פתרון מבחנים - מבני נתונים

11 ביולי 2025

בקובץ הבא כל הפתרונות למבחני מבני נתונים של גלעד, טליה איילת ושמואל מהדרייב וכן מבחנים מתל אביב. שימו לב שיש מצב שנפלו טעויות ולכן לא לקחת את התשובות כאן כמובן מאליו. מקווה שזה יעזור לשנים הבאות!

שאלה 4 ממבחן כלשהו

נגדיר k-יה אריתמטית כקבוצה סדורה עם k איברים טבעיים כך שההפרש בין כל שני מספרים זהה לאיזשהו קבוע c.
נתון מערך A בעל n מספרים שלמים שונים זה מזה.
א. הציעו אלגוריתם שמקבל כקלט מערך A ומס' k ובודק האם במערך A קיימת k-יה אריתמטית ושקר אחרת (כלומר האם קיימת קבוצה כזו שתקיים הדרוש). האלגוריתם צריך לרוץ בזמן ריצה של $O(n^k)$ במקרה הגרוע ביותר.אין צורך להוכיח נכונות אך הסבירו מדוע עובד. למשל עבור מערך $[10,15,20,12,30]$ ו $k=3$, האלגוריתם צריך למצוא את $[10,20,30]$ שהיא 3-יה ,ובמידה זהה גם את $[10,15,20]$. כלומר אין הכרח שהאיברים יהיו זה ליד זה, ויתכנו כמה k-יות.(13 נק')
ב. הציעו אלגוריתם אחר אשר פותר את אותה הבעייה כמו בסעיף הקודם, אך בזמן ריצה של $O(kn^2)$ בתוחלת , הסבירו מדוע עובד ונתחו את זמן הריצה שלו. ניתן להניח כי קיימת פונקציית גיבוב H טובה אשר מקבלת $\mathbb{N}\times\mathbb{N}$ וממפה אותם לטבלת גיבוב ב$O(1)$. (12 נק')
פתרון:
א. כמה k-יות יש בתוך מערך כלשהו? כלומר לבחור k איברים מתוך n איברים -

$$\binom{n}{k}=\frac{n!}{k!(n-k)!}=\frac{(n-k)!*(n-k+1)*....(n-1)n}{(n-k)!k!}=\frac{n(n-1)*.....(n-k+1)}{k!}\leq\frac{n^k}{k!}\leq n^k=O(n^k)$$

כלומר ברקורסיה נעבור על כל הקיות האפשריות.
בהינתן ק-יה, נעבור על כל האיברים ונבדוק האם ההפרש בין כל איברים קבוע, זה יעלה לנו $O(k)$. מכאן שסה"כ נקבל

$$k*n^k=O(n^k)$$

ב. מהרמז נעזר כמובן בטבלת האש. נבנה אותה וזה יעלה $O(n)$. יש לנו סה"כ n^2 זוגות איברים, לכן עבור כל זוג נכניס לטבלה את (a_i,a_j). נחשב את ההפרש בין השניים, נסמנו d, כעת אנחנו נרצה לבדוק האם הזוג (a_j,a_j+d) נמצא וכן הלאה, סה"כ k בדיקות כאלו ולכן

$$n^2*(k)=O(kn^2)$$

דרך נוספת - נכניס לטבלת האש את כל האיברים, לכל זוג נבדוק האם

$$a_i+d,a_i+2d,a_i+3d,..,a_i+k(k-1)$$

נמצאת במערך שלנו, זה יעלה k וכן לכל זוג (n^2זוגות) ולכן

$$O(kn^2)$$

חלק I

מבחן שמואל קליין - 1202

שאלה 1: להלן תיאור של מבנה נתונים המכיל מספרים טבעיים בין 1 ל m. הנתון m נקבע בפעולת האתחול של המבנה. המבנה תומך בפעולות הבאות

אתחול מקבוצה: מקבל פרמטר m וקבוצה $S \subseteq \{1, ...m\}$ בגודל n, כאשר $m = O(n^2)$ (כלומר המספר m יהיה עד לגודל הקבוצה בריבוע), זמן ריצה דרוש הינו $O(n)$.

הכנסת איבר חדש x בזמן $O(logn)$

הוצאת איבר קיים x בזמן $(O(logn)$

חיפוש איבר x בזמן $O(logn)$

תארו מבנה נתונים התומך בפעולות הנדרשות.

פתרון:

הקבוצה s הינה כבר ממוינת שכן מכילה מספרים טבעיים ברמה עולה, ולכן אנחנו הולכים לקחת אותה וליצור ממנה עץ AVL. זה יבטיח לנו הכנסה, הוצאה וחיפוש בתוך העץ ב$O(logn)$ כפי שאנחנו רוצים. כלומר מה שניתן הוא להסביר מדוע פעולת הבנייה תעלה $O(n)$. כיוון שאנחנו חסומים ע"י $m = O(n^2)$ וכן הקבוצה S היא מורכבת מהמספרים טבעיים $[1, ...m]$ אנחנו נוכל לבנות את עץ הAVL בצורה רקורסיבית. מה הכוונה? ראשית נמיין את מערך המספרים, כיוון שהוא חסום ב$1 \leq x \leq cn^2$ נוכל לבצע מיון ב$O(n)$. אח"כ ראינו בתרגול כי בהינתן מערך ממויין, נוכל לקחת את המערך הממויין שקיבלנו וכמו בתרגול לבנות מהמערך הממויין עץ AVL באופן רקורסיבי: בכל פעם ניקח את החציון ונפעיל ברקורסיה על שני האגפים השונים, נקבל שסיבוכיות זו תעלה לנו $O(n)$ בדיוק. מכאן שסה"כ מבנה הנתונים עובד כדרוש בזמן הדרוש.

שאלה 2:

נתון מערך של n מספרים שלמים A. כמו כן נתון כי לכל i מתקיים

$$A[i] - A[i+1] \leq 1$$

כלומר ההפרש בין כל שני איברים במערך, קטן בוודאות מ1. נניח כי n הוא חזקה של 2. נגדיר $a = A[1]$ וכן $b = A[n]$. נניח כי $a > b$. עליך לכתוב אלגוריתם יעיל אשר בהינתן $a \leq c \leq b$ ימצא אינדקס j כך ש$A[j] = c$.

פתרון:

הערה - לטובת הפשטות נניח בשאלה שמערך מתחיל מאינדקס 1, זה לא משנה. נתנו לנו רמז חזק עם n הוא חזקה של 2. אנחנו הולכים לבצע חיפוש בינארי באמצעות קפיצות של 2 במערך. ראשית, אם איכשהו $c = a$ או $c = b$ נחזיר 1 או n בהתאמה. מהנתון כי ההפרש בין כל שני איברים הוא קטן שווה 1, בוודאות קיים איבר c שכזה בתוך המערך. המוטיבציה תהיה להגיע לסיבוכיות של $O(logn)$, כלומר לא ממש דחוף לי לעבור על כל האיברים. נתחיל תמיד מאמצע המערך, אם האיבר c גדול מאמצא המערך נלך רקורסיבית לצד הימני ואם האיבר c קטן נלך לצד השמאלי. כך נפעל בכל פעם ובאופן רקורסיבי נמחק חצי מהמערך. נוכל לעשות זאת כיוון שהמערך הוא מונוטוני לא יורד, סה"כ נקבל כי

$$T(n) = T(\frac{n}{2}) + O(1) = O(logn)$$

טענה: נשים לב כי בוודאות קיים אינדקס כזה, שכן נניח בשלילה שלא ואז זה בסתירה לנתון מלמעלה.

שאלה 3:

הוכח באינדוקציה על מבנה העץ את הטענות הבאות על עץ טרינארי מושלם (לכל קודקוד יש 3 בנים)

1. לעץ טרינארי מושלם יש מס' עלים אי זוגי
2. לעץ טרינארי מושלם עם n עלים יש $\frac{n-1}{2}$ קודקודים פנימיים

פתרון:

1. נוכיח על גובה העץ. בסיס: $h = 0$ יש איבר אחד והוא הקודקוד הוא עלה יחיד ואכן מס' עלים 1 אי זוגי. צעד: נניח נכונות לעץ מגובה $h - 1$. יהי עץ טרינארי מושלם מגובה h. נתבונן בקודקוד r. מההגדרה יש לו 3 בנים:

$$a_1, a_2, a_3$$

גובה כל אחד מהם הינו עד $h-1$, וכן מס' העלים בהם אי זוגי. מכאן שסה"כ 3 כפול אי זוגי הוא אי זוגי ולכן בעץ שלנו יש מס' עלים אי זוגי.

2. נוכיח על מס' העלים בעץ. בסיס $n=1$ קודקוד יחיד אכן 0 פנימיים. נניח נכונות לעץ עם $n-1$ עלים. יהי עץ טרינארי עם n עלים. נקח את השורש r. יש לו שלושה בנים, כל אחד מהם יכיל לפחות עלה אחד, וכן מס' העלים שלהם קטן מn. לכן לפי הנחת האינדוקציה לכל אחד מהם יש $\frac{n_i-1}{2}$ עלים פנימיים, נשים לב כי $n_1+n_2+n_3=n$ וכן

$$\frac{n_1-1}{2}+\frac{n_2-1}{2}+\frac{n_3-1}{2}=\frac{n-3}{2}$$

, כמו כן נשים לב כי בספירה הזו לא כלנו את שורש העץ, שהינו גם עלה פנימי ולכן

$$\frac{n-3}{2}+1=\frac{n-1}{2}$$

כנדרש.

שאלה 4 - תכנות דינמי:

קלט: מספר N

פלט: ייצוג של N באמצעות מס' מינימלי של ריבועים. כל מס' טבעי יכול להיות מיוצג ע"י סכום ריבועים שכן $1^2=1$.

א. הציגו נוסחה רקורסיבית המחשבת את המס' המינימלי של הריבועים שסכומם הוא N. יש לפרט כל פרמטר בנוסחה את תפקידו, להסביר פלט ונכונות הנוסחה - לא צריך פורמלי.

ב. נתחו זמן ריצה ומקום של אלגוריתם תכנון דינמי כפונקציה של N.

ג. תארו בקצרה את התוספת הדרושה באלגוריתם למציאת אחד הפתרונות האפשריים.

פתרון:

תכלס זה קשה בטירוף אז בואו ננסה לפרק את זה לאט ובזהירות. זה מתמטיקה ואנחנו עפים על מתמטיקה. סכום ריבועים מקסימלי שיהיה הוא תמיד N, שכן לכל $N\in\mathbb{N}$ מתקיים פשוט

$$N=\sum_{n=1}^{N}1^2$$

עם זאת נראה כי גם ניתן לייצג ע"י $4=2^2$. כלומר תובנה ראשונה, סכום ריבועים מינימלי לכל מס' שקטן מ-4, יהיה תמיד המספר עצמו. כלומר למספרים $1,2,3$ סכום הריבועים המקסימליי זה לחבר 1^2. עבור 4, מתקיים פשוט 2^2 לכן מס' מנימלי של ריבועים הוא 1. ועבור 5? זה 4^2+1, כלומר 2 ריבועים. איך זה מקדם אותנו מכאן.

נשים לב כי אם $N=0$ אזי סכום הריבועים יהיה פשוט אפס, אם $N=1$ אזי סכום הריבועים יהיה פשוט אחד. אחרת, נתבונן בנוסחה הבאה:

$$f(k):=\left\{\begin{array}{cc} 0 & N=0 \\ min_{1\leq i\leq\sqrt{k}}\{1+f(k-i^2)\} & o.w \end{array}\right\}$$

מדוע זה עובד? נסתכל על 5 למשל, יש שתי אפשרויות:

$$5=1^2+1^2+1^2+1^2+1^2$$

$$5=1^2+2^2$$

מה צריך לקחת? את המינימום מבין השניים. כאשר כמובן הi שירוץ יהיה חייב להיות עד השורש, שכן $(\sqrt{k})^2=k$, לכן בכל פעם אנחנו נקח 1 בוודאות שיהיה הi שלנו, ואז נרוץ ונחפש את המינימום שיחזיר $k-i^2$.

מה הסיבוכיות של החרא הזה? לכל k אנחנו נעבור על $\sqrt{K}$ אפשרויות, ונאחסן בתוך מערך (לכן סיבוכיות המקום תהיה $O(n)$), אח"כ יש לנו מערך בגודל N, נרצה לחשב את הסיבוכיות הכוללת:

$$T(N) = \sum_{k=1}^{N} \sqrt{K} = N\sqrt{N}$$

גם מהגיון, הרי עשינו N פעמים $\sqrt{N}$. מכאן שסה"כ סיבוכיות תעלה לנו $O(n^{1.5})$ זמן ומקום $O(n)$, הגענו לסיבוכיות לינארית ולכן אנחנו מרוצים וסיימנו.

איך נמצא את הפתרון עצמו? בכל פעם שנמצא פתרון טוב יותר, נזכור לנו איזה ריבועים לקחנו, כלומר בהינתן תא $F[N] = x$, נלך שמאלה במערך עד שנמצא את תא $F[N - i^2] = x - 1$ כאשר נמצא כזה נשמור את i^2 שהרי הוא מהווה את החלק מהפתרון שלנו. כך נשמור ונזכור ובסוף נחזיר את הפלט.

חלק II

מבחן שמואל קליין וגלעד אשרוב - 3202 מועד ב

שאלה 1:

תהא A ערימת מקסימום בגודל n. יהי z צומת בערימה בעומק k. (הערה - עומק השורש הינו אפס, עומק קודקוד אחר הוא +1 מעומקו של ההורה). מוסיפים לכל אחד מאיברי תת הערמה של A המושרשרת בz את הקבוע $c > 0$. ברצוננו לתקן את המבנה כך שיחזור להיות ערימת מקסימום (ללא שינוי יתר הערכים במבנה A, אלא רק בתת העץ המושרש בz)

א. כתוב אלגוריתם לתיקון המבנה A שזמן ריצתו $O(klogn)$, הסבירו נכונות (אין צורך להוכיח)

ב. כתבו אלגוריתם לתיקון המבנה A שזמן ריצתו הוא $O(2^{log(n-k)}logn)$. הסבירו את נכונות האלגוריתם.

ג. לכל תת סעיף , איזה אלגוריתם מסעיף א או ב' עדיף אסימפטוטית מבחינת סיבוכיות זמן ריצה בכל אחד מהמקרים הבאים:

1. k קבוע
2. $k = loglogn$
3. $k = \frac{3}{4}logn$

פתרון:

א. ברגע שקרה מה שקרה, נשים לב כי תכונות הערימה יכלו להרס: כלומר, אנחנו יודעים שבערימה יש שתי דרישות: עץ בינארי כמעט שלם, ושהבנים יהיו קטנים מהאבא. עץ בינארי כמעט שלם נשמר, כיוון שלא הוספנו או הורדנו איברים למבנה, אלא רק הגדלנו ערכים של תת עץ מסויים בתוך הערימה. כלומר - עלינו לפתור את הבעיה ולדאוג שכל הבנים יהיו קטנים מהאבא: מה זמן הריצה אומר לנו? הוא אומר לבצע k פעמים $logn$. כלומר, אנחנו יודעים שבערימה למחוק ולהכניס איבר עולה $logn$, וצריך לבצע משהו כזה k פעמים. מעניין. כמו כן נשים לב שאיך נפגעו תנאי הערימה? מלמטה למעלה. נשים לב כי תת העץ המושרש z הינו עדיין ערימה, שכן היה ערימה לפני והגדילו כל תא בו במס' קבוע ולכן הוא נשמר ערימה. איפה החוסר איזון קורה? בנקודה המחברת את z אל שאר העץ. כלומר יתכן כי כעת z הוא קטן מאבא שלו. לכן אנחנו הולכים לבצע כך: אנחנו נקח את האיבר z ונפעפע אותו כלפי מעלה, בכל שלב אנחנו נבדוק האם הוא קטן מאבא שלו. אם כן נסיים ואם לא נחליף. הבדיקות האלו עולות $O(1)$ ובמקרה הגרוע ביותר נצטרך ללכת עד לראש העץ ב$O(logn)$. כמה חילופים כאלו יתכן שנצטרך לבצע? המקרה הגרוע הוא חילוף עבור כל רמה, מצב בו ממש נוצר לנו k $heapify$ פעמים על המסלול מz עד לשורש, כל $heapify$ עולה $logn$ ויש k מעברים כאלו במקרה הגרוע ולכן סה"כ $O(klogn)$.

ב. סתם ניסו לבלבל אותנו. $2^{log_2(n-k)} = n - k$. כלומר דרישת הסיבוכיות הינה

$$O((n-k)logn)$$

מה הדרישה כעת? לבצע $n-k$ $heapify$ כלשהם כנראה. מה זה $n-k$ זו השאלה. ערימה היא עץ בינארי כמעט שלם, בפרט אם מסתכלים עליה עד הרמה k היא מתנהגת ממש כמו עץ בינארי שלם. כלומר, עד לרמה הk יש $2^{k+1} - 1$ קודקודים. מכאן שבעץ הכולל יש בהינתן הגובה $logn$ יש $2^{logn+1} - 1 \approx n - 1$ קודקודים, ולכן מרמה k ואילך ומטה יש

$$n - 1 - 2^{k+1} + 1 = n - 2^k$$

קודקודים בערך.

נרצה להוציא את כל האיברים שבתת העץ שמושרש, כמה איברים כאלו יש? חסם של $n-2^k$. להוציא כל אחד מהם יעלה $logn$, נראה כי $2^k > k$ ולכן $-2^k < -k$ ולכן $n-2^k < n-k$. אנחנו נוציא אותם, זה סה"כ יסתכם בחסם מלמעלה של $n-k$ איברים, וכן אנחנו נזכור כי להוציא עולה $logn$ ולכן סה"כ חסם של $(n-k)logn$. כעת נחזיר אותם לעץ, הכנסה עולה גם $logn$ וכן חסם של אותו מספר, סה"כ נקבל $2(n-k)logn$, שזה אכן בסיבוכיות

$$O((n-k)logn))$$

סעיף ג:

א. אם k הינו קבוע, האלגוריתם הראשון יפעל ב$O(logn)$ זמן וכן השני יפעל ב$2^{logn}logn = nlogn$, לכן וודאי שעדיף הראשון.

ב. אם $k = loglogn$, האלגוריתם הראשון יפעל ב$O(loglogn * logn)$, בזמן שהשני יפעל בזמן של

$$O((n-k)logn) = (n - loglogn) * logn) = nlogn - loglogn * logn = O(nlogn)$$

ועדיין עדיף הראשון.

ג. אם $k = clogn$ שהרי שלושת רבע זה קבוע אז נקרא לו c , האלגוריתם הראשון יפעל ב$lognlogn$ וכן השני ב$(n - logn)logn = nlogn - lognlogn$ ולכן עדיף הראשון.

שאלה 3:

א. הצע מבנה נתונים שתומך בפעולות הבאות:
1. $Push(s, x)$ - הכנסת האיבר x אל s
2. $Pop(x, s)$ - מחיקת האיבר האחרון שנכנס לs
3. $Add(s, d)$ - הוספת הערך d ממשי לכל איברי המבנה s
4. $Small(s)$- החזרת הערך המינימלי בs.

כל אחת מהפעולות צריכה להיות ב$O(1)$

ב. הוכח הפרך - ניתן להוסיף פעולה שמוחקת את המינימלי ב$O(1)$.

פתרון:

א. הרי ברור שכל הפעולות קלות לפתרון שכן אפשר להשתמש במחסנית להכניס ולהוציא איבר אחרון, וכן ניתן לדעת את המינימלי ביותר שכן ניתן להוסיף שדה לכל איבר שיציין את המינימלי ביותר בעת ההכנסה, שכן בכל פעם נבדוק האם האיבר החדש קטן מהמינימלי ואם כן נעדכן. כל זה יקרה ב$O(1)$. הפעולה הבעייתית הינה add.

נלך על הגישה הבאה: הרי אף אחד לא יודע כרגע כיצד נראית המחסנית שלי. אנחנו הולכים להוסיף add ברגע נתון ביציאה מהמחסנית בלבד. אנחנו הולכים לשמור משתנה בשם x שאליו יכנס כמות ההוספות וההכנסות שלנו. למשל, אם יש סדרה של פעולות add עם ההוספות $d_1, d_2, ...d_n$ אזי לאחר הסדרה הזו נקבל כי $x = \sum_{i=1}^{n} d_i$. נראה כי בכל רגע נתון, לאחר פעולת add_i האיבר המינימלי x_{mn} ישאר האיבר המינימלי שהיה קודם לכן, שכן אנחנו מבצעים הגדלה של כל איברי המערך באותו ערך. רק שאנחנו נצטרך לעדכן את המינימלי עם הערך d_i. כלומר סה"כ אנחנו נתאר את הפעולות הבאות:

הכנסה - בדיוק כמו שמכניסים למחסנית רגילה. אנחנו נשמור שני אינדקסים בהכנסה שאחד יציין את הערך המינימלי ביותר עד לאותו רגע נתון וכן השני יהיה הערך x בזמן ההכנסה (מדוע? בהינתן סדרה של פעולות הגדלה ולאחריהן הכנסה אנחנו לא נרצה כי האיבר יקבל את התוספות שהיו לפני שהוא נכנס למחסנית.)

הוצאה - אנחנו הולכים למחוק איבר אחרון כמו שעושים במחסנית רגילה, אנחנו צריכים גם להחזיר אותו אז אנחנו נחזיר את הערך שלו + הערך של x פחות הערך d_i ששמרנו כשהיה שהוא נכנס (וזה סה"כ השינוי תוספת שהיה לו)

הוספה - כפי שתיארנו מעלה אנחנו הולכים להוסיף את הסכום הזה לכולם בגישה טריקית ולא בכל רגע נתון כולם יגדלו, אלא רק בזמן ההוצאה של איברים או החזרתם. אנחנו נעדכן לכל איבר את הערך d_i שלו שיציין מה יהיה הערך x בזמן ההכנסה, וכן בכל פעולת add אנחנו נעדכן את x. בסוף נחזיר עבור איבר שנרצה את $x - d_i$

החזרת מינימום - בזכות הערך הנוסף ששמרנו בתוך המחסנית, אנחנו נדע בכל רגע נתון מיהו האיבר המינימלי. בהינתן שנצטרך להחזיר אותו כעת, אנחנו נחזיר אותו ועוד הערך של איקס, פחות הערך שהיה לd_i שלו ברגע ההכנסה. כיצד נדע מיהו הערך d_i שלו ברגע ההכנסה? אז כרגע נחליט שנשמור גם את הערך הזה כערך שלישי ביחד עם כל הערכים ששמרנו עד עכשיו, זה יציין את הערך d_i שהיה לאיבר המינימום בעת ההכנסה. וסהכ בסוף נחזיר את $min + x - min(d_i)$.

ב. הפרכה: לא יתכן שנוכל להוסיף פעולה שתמחק את המינימום ב$O(1)$. אם המינימום הוא בראש המחסנית - בסדר. ניתן לעשות זאת. אבל מה אם איננו? נב"ש שכן היה אפשר להוסיף פעולה x שמוחקת את המינימום

ב$O(1)$. בהתאם למבנה שבנינו בסעיף הקודם, ניתן לדעת מה הערך של איבר המינימום בלבד. הפעולה הייתה צריכה לחפש אותו במחסנית, לשלוף את האיברים למחסנית נוספת, להוציא אותו ואז להחזירם. במקרה הגרוע ביותר האיבר המינימלי ביותר היה נמצא בתחתית המחסנית והיינו צריכים לשלוף $n-1$ איברים, להוציא אותו ואז להחזירם. זה סה"כ $O(n)$. גם אם היינו מחזיקים $pointer$ לאיבר המינימלי, היינו זקוקים להוציא את כל האיברים שלפניו מהמבנה. כלומר, תמיד בשביל להוציא איבר כלשהו מהמחסנית נזדקק להוציא את כל אלו שלפניו ממנה, ובמקרה הגרוע זה $O(n)$.

דרך נוספת להפריך: נב"ש שכן ניתן לעשות זאת. אזי נבצע n פעולות הצעת איבר המינימום ונשמור את הנתונים במערך. קיבלנו מערך ממויין ב$O(n)$ עבודה, בסתירה לכך שמיון מבוסס השוואות מקבל חסם $\Omega(nlogn)$.

שאלה 4:

א. מערך A הוא מפותל אם

$$A[1] \leq A[2], A[2] \geq A[3], A[3] \leq A[4], ...A[2n-1] \leq A[2n], A[2n] \geq A[2n+1]$$

בהינתן מערך B לא ממויין של מספרים, הציעו אלגוריתם יעיל ביותר שמוצא פרמוטציה $A[1, ...2n+1]$ של B כך שA הוא מפותל.

פתרון: השאלה לא כזו ברורה, אז בתכלס מה שצריך הוא למצוא איזשהו סדר כך שהמערך B יהיה מערך מפותל. הפתרון הלא יעיל בעליל הוא לעבור על כל הפרמוטציות האפשריות $n!$ ולסרוק אותם ולקבל שזה $O(n!)$. גועל נפש וממש לא הכוונה.

נשתמש באלגוריתם סלקט על האיבר האמצעי בגודלו במערך, כלומר על n+1. נשים לב שלאחר השימוש באלגוריתם נקבלn איברים שקטנים מהאיבר האמצעי שנמצאים בצד שמאל של המערך ו n איברים שגדולים מהאיבר האמצעי בצד ימין של המערך. כעת ניצור מערך חדש בגודל $2n+1$ ונשים את כל האיברים שבחצי השמאלי של המערך B באיברים האי זוגיים במערך החדש, ואת כל החצי הימני באיברים הזוגיים במערך החדש. סה"כ $O(n)$.

ב. מערך של מס' הוא k זוגי מעורבב כאשר קיימים בדיוק k מספרים זוגיים בA והאי זוגיים הינם ממויינים. בהינתן מערך k זוגי מעורבב המכיל n מספרים שונים וכן $k = \frac{n}{logn}$, הצע אלגוריתם שרץ ב$O(n)$ בשביל למיין את A.

פתרון: קיימים $n - \frac{n}{logn}$ מספרים אי זוגיים. כעת המוטיבציה תהיה לבדוק האם בהינתן $m = n - \frac{n}{logn}$ ניתן לקבל סיבוכיות מעניינת. מהו $log(m)$? בואו נבדוק

$$log(m) = log(n - \frac{n}{logn}) = log(n(1 - \frac{1}{logn})) = logn + log(1 - \frac{1}{logn}) \leq log(n)$$

נעבור על האיברים האי זוגיים הממויינים במערך A ונכניסם לתוך מערך B בגודל m. סה"כ נקבל מערך ממויין של אי זוגיים. באופן דומה ניצור מערך C שיכיל $\frac{n}{logn}$איברים, ונעבור שוב ונכניס את הזוגיים לתוך המערך. נרצה למיין כעת את המערך C שכן איננו ממויין. נמיינו בדרך כלשהי (ערימה או קוויק סורט) וזה יעלה

$$\frac{n}{logn} log(\frac{n}{logn}) = \frac{n}{logn}(logn - loglogn) \leq \frac{n}{logn} * logn = O(n)$$

כלומר למיין את המערך הזה החדש C יעלה לנו סה"כ $O(n)$. כעת קיבלנו מערך B שמכיל את כל האיברים האי זוגיים הממויינים, ומערך C שמכיל את הזוגיים. כעת ניצור מערך A חדש נקי באותו גודל. בכל שלב נשאל מי יותר גדול? האיבר הראשון במערך B או האיבר הראשון במערך C, נכניס את הקטן יותר בכל פעם ואם התקדמנו נקדם את האינדקס של השאילתא. סה"כ עברנו על $2n+1$ איברים כעת.

מה סיבוכיות המיון? ובכן להכניס את האי זוגיים עלה n, להכניס את הזוגיים גם כן n, למיין עלה n, לאחד עלה $2n+1$, סה"כ עלה $4n+1$, ולכן $O(n)$.

מבחן 8102 טומי קליין מועד א'

שאלה 1

הבעיה: ברצוננו לנהל אוסף של n מספרים שונים ע"י בניית מבנה נתונים בזמן $O(n)$ כך שנוכל אח"כ לבצע כל אחת מהפעולות בזמן ממוצע של $O(logn)$: הוצאת האיבר המקסימלי, הכנסת הערך k, הוצאת הערך k.

א. מדוע השימוש בערימה אינו עונה במלואו על הדרישות?
ב. הצע מבנה נתונים שכן יכולים לתמוך בכל הנדרש.

פתרון: א. אכן ניתן לבנות ערימה ב$O(n)$ ממערך למשל, ניתן להוציא את המקסימלי ב$O(logn)$ וכן ניתן גם להכניס ערך ב$O(logn)$. מה לא ניתן? להוציא ערך k. לחפש בתוך ערימה זה בלתי אפשרי, היא לא מסודרת בשום סדר שמאפשר חיפוש בינארי ולפני שמוציאים איבר צריך לחפש אותו, לכן בוודאות אנחנו נצטרך במקרה הגרוע לעבור על כל הערימה ב$O(n)$ לפני שנוכל למחוק. מכאן שזה בלתי אפשרי ב$O(logn)$.

ב. אנחנו יודעים לבנות ערימה ב$O(n)$. נבנה ערימת מקסימום, משם להוציא את המקסימום זה קל וגם להכניס ערך k זה לא בעיה ב$O(logn)$. מה שבעייתי הוא הוצאת הערך k. אמרו ממוצע לכן נבנה טבלת האש בנוסף, לבנות טבלה יעלה לנו $O(n)$. הטבלה תחזיק פוינטרים לאיברים שבערימה. כעת, אנחנו נחפש בטבלת האש את הערך k, זה יעלה לנו בתוחלת $O(1)$, ואז נלך לפוינטר שיביא אותנו למיקום המדויק ביותר בערימה של אותו איבר, נמחק אותו (כלומר נחליף אותו עם האיבר שנמצא ברמה התחתונה מימין - אנחנו בעץ כמעט שלם), ואז האלגוריתם זהה למחיקה מערימה רגילה - גלגולים ופעפוע כלפי מעלה ב$O(logn)$. נעשה זאת פורמלי יותר -

בנייה: נבנה ערימה ב$O(n)$ וכן טבלת האש ב$O(n)$. כל איבר שבטבלת האש יחזיק פוינטר לאותו הערך בתוך הערימה. כמו כן נחזיק פוינטר גם מהערימה לטבלת האש. זה יהיה פוינטר דו כיווני.

מחיקת איבר המקסימום: ניתן לעשות זאת הן בערימה והן בטבלת האש ב$O(1)$, כאשר נמחק מטבלת האש את המקסימלי רגע לפני נלך עם הפוינטר למחוק אותו מטבלת האש. יעלה $O(logn)$.

הכנסת ערך k: נכניס אותו לערימה רגיל, וגם לטבלת האש באופן רגיל, ונחבר עם פוינטרים. יעלה $O(logn)$

הוצאת הערך k: נחפש את האיבר בטבלת האש ב$O(1)$. הפוינטר ישלח אותנו למיקום בערימה. נגיע אליו ונחליף עם זה שנמצא ברמה התחתונה מימין, ונפעפע כרגיל. כמובן שנמחק אחכ גם מטבלת האש ב$O(1)$ בתוחלת. סה"כ $O(logn)$.

שאלה 2

מיון מהיר פועל ע"י 1. בחירת איבר חלוקה K (מטעמי עצלנות נבחר לרוב כ$A[1] = k$)
2. חלוקת n האיברים במערך ל2 קבוצות s_1, s_2 שמכילות בהתאמה את האיברים הקטנים מk והגדולים מk. ואז סידור המערך מחדש לs_1ks_2
3. המשך רקורסיבי עד למיון מלא

עליך להוכיח בשאלה כי מיון מהיר הוא אופטימלי בממוצע. כאשר בכל השאלה הממוצע מתייחס לכך שהמיקום בסידור הממויין של איבר החלוקה יכול להיות כל אחד מn המקומות בהסתברות שווה.

אנחנו נדריך אותך בשאלה כיוון שזו שאלה קשה:

0. נגדיר $T(n)$= מס' השוואות ממוצע בכדי למיין n איברים ע"י QS
1. $T(n) = a(n) + b(n)\sum_{i=1}^{n} c(i)$ כאשר $a(n) = n + 1$ הוא מס' ההשוואות לביצוע החלוקה, $b(n) = \frac{1}{N}$ היא ההסתברות שאיבר החלוקה יפול למיקום i ו$c(i)$ היא העלות ההמשך הרקורסיבי במקרה שאיבר החלוקה יפול למיקום i. עליכם למצוא את $c(i)$ במונחים של T . רשום את הנוסחה הרקורסיבית שמתקבלת עבור $T(n)$
2. הכפל את שני צידי המשוואה בn בשביל לטפל רק במספרים שלמים
3. נסחו משוואה דומה אך שונה עם פרמטר $n - 1$ במקום n
4. צרו משוואה חדשה ע"י חיסור אגף אגף של המשוואות הקודמות וסידור מחדש
5. חלקו שני אגפים ב$n(n + 1)$
6. הגדירו $H(n) = \frac{T(n)}{n+1}$ וקבלו נוסחה רקורסיבית עבור $H(n)$
7. פתרו עבור $H(n)$ והסיקו ל$T(n)$. מה המסקנה באשר לאופטימליות של $QUICKSORT$?

פתרון:

נלך אחרי השלבים ונקווה לטוב. נגדיר $T(n)$ מס' ההשוואות הממוצע בשביל למיין n איברים ע"י QS.

מהנוסחה נראה כי

$$T(n) = n + 1 + \frac{1}{n}\sum_{i=1}^{n} c(i)$$

עלינו להבין מיהו $c(i)$. אם איבר החלוקה נופל למיקום i, אזי אנחנו צריכים לטפל בשני תתי בעיות: i-1, ו$n - i$. לכן

$$T(n) = n + 1 + \frac{1}{n}\sum_{i=1}^{n} T(i - 1) + T(n - i)$$

$$\sum_{i=1}^{n} T(i-1)+T(n-i)=T(0)+T(n-1)+T(1)+T(n-2)+....+T(n)+T(0)=2\sum_{i=1}^{n} T(n-i)$$

כלומר,

$$T(n)=n+1+\frac{2}{n}\sum_{i=1}^{n} T(n-i)$$

כעת לפי ההדרכה נכפיל את שני הצדדים בn:

$$T(n)*n=n^2+n+2\sum_{i=1}^{n} T(n-i)$$

כעת נרצה לנסח משוואה עם $n-1$.

$$T(n-1)*(n-1)=(n-1)^2+n-1+2\sum_{i=1}^{n-1} T(n-1-i)$$

ניצור משוואה חדשה ע"י חיסור שתי המשוואות הקודמות:

$$T(n)n-T(n-1)*(n-1)=n^2+n+2\sum_{i=1}^{n} T(n-i)-[(n-1)^2+n-1+2\sum_{i=1}^{n-1} T(n-1-i)]$$

$$T(n)n-T(n-1)*(n-1)=n^2+n-(n-1)^2-(n-1)+2\sum_{i=1}^{n} T(n-i)-2\sum_{i=1}^{n-1} T(n-1-i)$$

$$T(n)n-T(n-1)*(n-1)=n^2+n-n^2+2n-1-n+1+2\sum_{i=1}^{n} T(n-i)-2\sum_{i=1}^{n-1} T(n-1-i)$$

$$T(n)n-T(n-1)*(n-1)=2n+2[\sum_{i=1}^{n} T(n-i)-\sum_{i=1}^{n-1} T(n-1-i)]$$

$$\sum_{i=1}^{n} T(n-i)-\sum_{i=1}^{n-1} T(n-1-i)=T(n-1)+T(n-2)+....+T(0)-[T(n-2)-T(n-3)-....T(0)]=$$

$$=T(n-1)$$

כלומר סה"כ המשוואה החדשה תהיה

$$T(n)n - T(n-1)*(n-1) = 2n + 2T(n-1)$$

נעביר אגף

$$T(n)n = 2n + 2T(n-1) + T(n-1)*(n-1)$$

$$T(n)*n = 2n + T(n-1)(2+n-1)$$

$$T(n)*n = 2n + T(n-1)*(n+1)$$

כעת ההנחיה היא חלקו שני אגפים ב$n(n+1)$ וכך נעשה־

$$\frac{T(n)}{n+1} = \frac{2}{n+1} + \frac{T(n-1)}{n}$$

נגדיר $H(n) = \frac{T(n)}{n+1}$ ונקבל

$$H(n) = \frac{2}{n+1} + H(n-1)$$

ננסה כעת להבין מה הפתרון של נוסחת הנסיגה הזו

$$H(n-1) = \frac{2}{n} + H(n-2)$$

$$H(n-2) = \frac{2}{n-1} + H(n-3)$$

נציב

$$H(n) = \frac{2}{n+1} + H(n-1) = \frac{2}{n+1} + \frac{2}{n} + H(n-2) = \frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + H(n-3) =$$

$$\frac{2}{n+1} + \frac{2}{n} + \frac{2}{n-1} + \frac{2}{n-2} + ... + 2$$

סה"כ נשים לב כי נקבל את הנוסחה הבאה:

$$H(n) = \sum_{i=0}^{n-1} \frac{2}{n+1-i}$$

זהו טור הרמוני.
נקבל כי לפי טור הרמוני, $H(n) = O(logn)$. מכאן המסקנה היא כי $H(n) = \frac{T(n)}{n+1}$ ולכן $logn(n+1) = T(n)$ כלומר $T(n) = O(nlogn)$. אכן הגיוני, בהתאם לחסם תחתון שלמדנו בהרצאה. מש"ל.

שאלה 3

השאלה: יהי D מבנה נתונים כלשהו המנהל קבוצה של n איברים ותומך בפעולות של הוצאת האיבר הקטן ביותר, ובפעולת הכנסת איבר. הפעולות מתבצעות ע"י סמך השוואות בין איברים. הוכח או הפרך את הטענה הבאה: לפחות אחת מהפעולות הנ"ל (הוצאת min או הכנסה) דורשת זמן $\Omega(logn)$

פתרון: הוכחה: נניח בשלילה כי אף אחת מן הפעולות לא דורשת זמן $\Omega(logn)$. אזי, ניתן להוציא את האיבר המינימלי בפחות מ$\Omega(logn)$. נבצע כך n פעמים ־ ונקבל כי ביצענו מיון מבוסס השוואות בפחות מ$nlogn$. בסתירה. ובפורמליות יותר ־ נבצע n הכנסות למערך בזמן שקטן מ$\Omega(logn)$. כלומר ב$o(logn)$. אח"כ נבצע n הוצאות ב$o(logn)$. קיבלנו מיון ב$o(nlogn)$ בסתירה לכך שמיון מבוסס השוואות הוא ב$\Omega(nlogn)$. כנדרש.

שאלה 4:

ענה נכון או לא נכון על ההבאים. מותר אך לא חובה לכתוב הסבר של עד שורה לכל סעיף:

א. לכל n קיים עץ חיפוש בינארי שגובהו $O(\sqrt{n})$

פתרון: לכל n קיים עץ avl שגובהו $O(logn)$ ואכן הוא גם בינארי, ככה שהטענה נכונה

ב. ברשימה מקושרת באורך n, כדאי לשמור גם מצביע לאיבר האמצעי כי בהשקעה של $O(1)$ שטח קיבלנו אפשרות לבצע חיפוש בינארי

פתרון: הטענה לא נכונה, שכן זה לא מערך ולא ניתן לקצץ את החיפוש כיצד שרוצים. במערך זה היה נכון

ג. בטבלת האש בגודל 2500 הוכנסו 1000 איברים ע"י שימוש ב$Hashing-uniform$. הזמן הממוצע לחיפוש מוצלח כאשר לכל איבר שהוכנס הסתברות שווה להופיע בחיפוש היא $\frac{5}{2}ln(\frac{5}{3})$

פתרון: הנוסחה היא $\frac{1}{a}ln(\frac{1}{1-a})$, אם נציב $a=\frac{1000}{2500}=\frac{2}{5}$ אכן נקבל

$$\frac{5}{2}ln(\frac{1}{1-\frac{2}{5}})=\frac{5}{2}ln(\frac{1}{\frac{3}{5}})=\frac{5}{2}ln(\frac{5}{3})$$

ד. נתון $B-tree$ מדרגה m שמכיל k איברים. אם נכניס איבר נוסף לעץ ובשלב הבא נמחק אותו, העץ שמתקבל יהיה זהה לעץ שממנו יצאנו

פתרון: לא נכון. ראינו בתרגול ובתרגילי בית כבר שזה חרטא.

ה. החסם התחתון למיזוג 2 סדרות ממוינות באורך n והשנייה באורך 3 הוא $n-2$

פתרון: עלינו למזג לרשימה ממוינת אחת. נצטרך לעבור לפחות על $n+1$ איברים לפחות

ו. לכל n טבעי קיים עץ AVL המכיל n קודקודים שסריקת ה$post-order$ שלו היא סדרה ממוינת

פתרון: אין לי שמץ

ז. בכל רשימה מקושרת, קיים איבר אליו ניתן לגשת ב$O(1)$ זמן

פתרון: כן, לאיבר הראשון.

ח. לכל n טבעי קיים עץ בינארי המכיל בדיוק n עלים ו$2n$ קודקודים פנימיים

פתרון: נכון

2202 מועד א שמואל טומי קליין

שאלה 1

השאלה:

א. נתונה קבוצה של n מספרים, הראו בשתי דרכים שונות כיצד ניתן לבנות בזמן $O(nlogn)$ עץ חיפוש בינארי שבקודקודיו שמורים הערכים שבתוך הקבוצה. בדרכים שונות ־ כלומר עם רעיונות שונים ולא רק שינוי הפרמטרים.

ב. הראו שאי אפשר לבנות את עץ החיפוש הבינארי מהסעיף הקודם בזמן $o(nlogn)$.

פתרון:

א. פתרון ראשון יהיה בדרך המיון ־ אומרים $nlogn$ אז נמיין את המערך בסדר עולה. זה יעלה $nlogn$ לפי סוגי מיון שראינו בהרצאה. אח"כ נבנה באופן רקורסיבי את העץ: נלך לערך האמצעי ביותר (בה"כ מס' האיברים אי זוגי אז נלך לאיבר $\frac{n+1}{2}$) נסמן אותו כשורש עץ החיפוש, ונפעיל ברקורסיה על שני הצדדים. נקבל את נוסחת הנסיגה $T(n)=2T(\frac{n}{2})+O(1)=O(n)$ ולכן סה"כ בניית העץ בדרך זו תעלה $n+nlogn=O(nlogn)$

פתרון שני יהיה בניית עץ AVL. הרי כל עץ AVL הוא עץ חיפוש בינארי. לשם כך נמיין את המערך שוב ב$nlogn$. אח"כ נבצע n פעמים את הפעולה הבאה: ניקח את האיבר הראשון במערך, נכניס אותו לעץ AVL וחוזר חלילה. ראינו בהרצאה כי הכנסה לעץ AVL תעלה $logn$. מכאן שn פעמים זה $nlogn$. סה"כ $nlogn+nlogn=O(nlogn)$

ב. עלינו להראות למעשה ש$\Omega(nlogn)$ זה חסם תחתון לבנייה. נניח בשלילה כי ניתן לבנות עץ חיפוש בינארי ב$o(nlogn)$. מכאן שיש לנו עץ חיפוש ולכן אפשר לחפש את האיברי המינימלי ביותר (שמאל למטה) ולהוציא אותו ב$logn$. סה"כ נבצע זאת $nlogn$ פעמים וקיבלנו מיון מבוסס השוואות ב$o(nlogn)$ בסתירה לכך שחסם תחתון הוא $\Omega(nlogn)$ כמו שראינו בהרצאה. דרך אחרת היא תהיה להדפיס ב$inOrder$ שזה יעלה $O(n)$ וסה"כ קיבלנו מיון מבוסס השוואות ב$o(nlogn)$, בסתירה.

שאלה 2 - תכנון דינמי

נתונה קבוצה של n ערים $a_1, ...a_n$. ומטריצה $C[i,j]$ עבור $1 \le i,j \le n$ שנותנת את עלות הנסיעה מעיר a_i לעיר a_j(ללא תחנות ביניים). אם אין אפשרות לנסיעה ישירה בין הערים אזי $c[i,j] = \infty$. נרצה לבנות מטריצה $D[i,j]$ כך שתתן את עלות הנסיעה הזולה ביותר בין a_i לבין a_j כאשר מותר להשתמש בתחנות ביניים (אין צורך לפרט מיהן, אלא רק את המחיר)

א. 13 נקודות - הגדירו נוסחה רקורסיבית $E[i,j,k]$ עבור $0 \le k \le n$ וכן $1 \le i,j \le n$ שתחזיר את עלות הנסיעה הזולה ביותר מa_i לa_j כאשר מותר להשתמש בתחנות ביניים מהקבוצה $\{1,..k\}$. יש לכלול תנאי עצירה בנוסחה.

ב. 6 נקודות - הגדירו את המטריצה D שברצוננו לבנות במונחים של הנוסחה הרקורסיבית $E[i,j,k]$

ג. מהי סיבוכיות הזמן והמקום הנדרשות עבור אלגוריתם התכנון הדינמי? נמקו. שימו לב - אין צורך לשחזר את המסלולים.

פתרון:

א. תכנון דינמי זה קשה לכן נעבוד לאט. מה אנחנו יודעים? אם $i = j$ אזי מדובר בנסיעה מעיר לעצמה, ייתכן. לכן במצב זה נחזיר את הערך 0. אחרת, אנחנו רוצים לרוץ על הקבוצה $\{1,..k\}$. אם $k = 0$ נחזיר את ערך הנסיעה בין הערים ללא ערים שכנות שכן מדובר בתחנה בודדת. אחרת, נחזיר את המינימום בין לא לקחת את התחנה לבין לקחת, נתאר זאת כך:

$$E[i,j,k] := \left\{ \begin{array}{cc} 0 & i=j \\ c[i,j] & k=0 \\ min\{E[i,j,k-1], E[i,k,k-1]+E[k,j,k-1] & o.w \end{array} \right\}$$

מה קורה כאן? אפשרות ראשונה היא לא לקחת את התחנה, ואפשרות אחת היא ללכת על המסלול עד לתחנה ולא לקחת אותה ועוד ממנה עד לj. כלומר המסלול עד לתחנה, והמסלול ממנה אל j.

ב. בכל איבר של המטריצה יהיה למעשה $E[i,j,n]$. כלומר סה"כ $D[i,j] = E[i,j,n]$. שכן זה n כיוון שרוצים לעבור על כל התחנות.

ג. נשים לב כי נרצה לבנות מטריצה תלת מימדית, שתכיל את כל ערכי k האפשריים וכן הi האפשריים והj האפשריים. לשם כך נהיה חייבים במטריצה תלת מימדית ולכן סיבוכיות המקום תהיה $O(n^3)$. כמו כן, נמלא אותה וגם זה יעלה לנו בזמן $O(n^3)$. סה"כ נעבור על n^3 תאים ונעשה פעולות של $O(1)$ וכן נעבור על מטריצה D $O(n^2$ פעולות, לכן אנחנו מדברים על סדר גודל של $O(n^{3)}$זמן.

שאלה 4

השאלה: הציגו אלגוריתם המקבל קבוע k וערימת מינימום בגודל n, המיוצגת במערך, ומחזיר רשימה ממוינת של k האיברים הקטנים במערך, מהקטן לגדול, בעלות זמן ריצה של $O(min\{k^2, klogn\})$ וסיבוכיות מקום נוספת של $O(1)$.

פתרון:

נראה כי לפתור זאת ב$klogn$ אנחנו יודעים. כלומר, נבנה ערימת מינימום $O(n)$. ונשלוף k פעמים את האיבר המינימלי - זה יעלה $klogn$. מכאן שסה"כ הסיבוכיות היא $O(klogn)$. אבל נשאלת השאלה מתי $k^2 \le klogn$. כאשר $k \le logn$, זה ייתכן כמובן. לכן אנחנו נתאר את האלגוריתם הבא: k האיברים הקטנים ביותר בערימה נמצאים עד רמה k בערימה. לשם כך אנחנו הולכים לחפש בכל פעם בערימה עד רמה k. מה שמתחת לא מעניין, כל פעולת הוצאת איבר המינימום תעלה $O(k)$ כגובה הערימה עליה אנחנו מסתכלים, וכן נבצע זאת k פעמים ולכן סה"כ זה k^2פעולות. מתי זה לא יהיה כך? אם $k = logn$, כלומר האיברים יכולים להיות עד הרמה האחרונה (הרי גובה ערימה הוא $logn$ במס' האיברים), ואז במקרה זה נקבל כי $klogn$ תהיה הסיבוכיות.

2202 מועד ב' טופס איילת בוטמן

שאלה 1

השאלה: כדי לזרז את החיפוש הבינארי במערך ממויין A בגודל n, הוצע להוסיף מערך B בגודל $logn$ שיכיל כל איבר $\frac{n}{logn}$ של A. כלומר, $A[i*\frac{n}{logn}]=B[i]$ לכל $1\le i\le logn$, לכן שני ערכים עוקבים בB מייצגים את תחום הערכים של קטע באורך $\frac{n}{logn}$ במערך A. החיפוש של איבר x יתנהל כעת כך:

א. חיפוש בB לקביעת הקטע המתאים בA

ב. חיפוש בינארי רק בקטע המתאים מהערך A למציאת המספר x.

1. האם שיטה זו עדיפה על חיפוש בינארי רגיל רק בA בהנחה שהחיפוש בB בשלב 1 נעשה על ידי חיפוש סידרתי (כי גודלו של B הרבה יותר קטן מA)?

2. האם שיטה זו עדפה על חיפוש בינארי רגיל רק בA, בהנחה שהחיפוש בB בשלב 1 נעשה ע"י חיפוש בינארי?

הפתרון:

א. בהנחה שאנחנו הולכים לבצע את החיפוש בB בחיפוש סידרתי, אנחנו הולכים לדבר על המקרה הגרוע ביותר בו נמצא את התחום רק בסוף כלומר לאחר $\frac{n}{logn}$פעולות. לאחר מכן יש לנו מקטע של $\frac{n}{logn}$איברים. אנחנו נרצה לבצע בקטע זה חיפוש בינארי. ידוע כי חיפוש בינארי מתבצע ב$log(k)$ כאשר k הוא אורך הקלט. במקרה שלנו $k=\frac{n}{logn}$ ולכן נבצע חיפוש בינארי שיעלה

$$log(k)=log(\frac{n}{logn})=logn-log(logn)$$

סה"כ החיפוש יעלה לנו $logn-log(logn)+\frac{n}{logn}$. זה לא משפר את הסיבוכיות האסימפטוטית, שכן זה נמצא בסיבוכיות של לפחות $O(logn)$, כמו חיפוש בינארי רגיל.

ב. בהנחה שאנחנו הולכים לבצע חיפוש בינארי גם בB, יעלה לנו שם $log(\frac{n}{logn})$ וכן יעלה אותו הדבר גם בתוך המערך כמו שראינו קודם לכן סה"כ החיפוש יהיה

$$2logn-2log(logn)=O(logn)$$

לצערנו שוב, אנחנו הולכים לקבל סיבוכיות שהינה $O(logn)$.

שאלה 3

השאלה: תהא ערימת מינימום בגודל n. ואיבר עם ערך x. בנוסף ידוע כי מס' האיברים בערימה הקטנים מx הוא k. תארו אלגוריתם שמשיב את כל האיברים הקטנים מx בזמן $O(k)$.

פתרון: זו נשמעת כמו שאלה קשה אז ננסה לאכול אותה לאט. בכל זאת 52 נקודות. מה יודעים? בערימה להוציא איבר זה $O(logn)$. קל לדעת את המינימום ב$O(1)$. כלומר על פניו אם נרצה להוציא k איברים נצטרך לעשות זאת ב$klogn$. איך נבצע זאת ב$O(k)$? נשים לב שמובטח לנו שיש k שקטנים ממנו. אנחנו יודעים שבערימת מינימום הבנים גדולים מהאבא וכן הלאה.

הרעיון הוא כזה: נתחיל את שלב 1 עם השורש.

1. אם האיבר שאנחנו נמצאים בו קטן מאיקס, נעבור לשלב 2. אחרת נחזור (למה? בוודאות יהיו גדולים ממנו אלו שמתחתיו)

2. תדפיס את האיבר, ואז תעבור לשלב 1 עם הבן הימני והבן השמאלי שלו.

נכונות: אנחנו יודעים כי בערימת מינימום הבנים גדולים שווים מהאבא. לכן אם האיבר הראשון לא קטן מאיקס, בוודאות כולם לא קטנים מאיקס. לכן אנחנו הולכים להתחיל בוודאות מהאיבר העליון.

סיבוכיות: נראה כי כיוון שמובטחים לנו k איברים שקטנים מאיקס, התהליך ימשך בדיוק k פעמים שכן פעולת הדפסה או מעבר לבן עולות $O(1)$. מכאן שנקבל סיבוכיות של $O(k)$. הערה. יתכן שנבקר בנודים שלא טובים אבל אז נעצור בהם, ולכן מובטח $O(k)$.

שאלה 4

בהינתן מספר טבעי n, ברצוננו לחשב את מס' הצעדים המינימלי להגיע למספר 1. מותר לבצע את שתי הפעולות הבאות: אם קיימים $a\le b$ כך ש $n=ab$, אזי נוכל להפחית את n להיות b. כמו כן, מותר תמיד להפחית את n להיות $n-1$. לדוגמה: עבור $n=100$ ניתן בחמישה צעדים להגיע ל1: $100\to 20\to 5\to 4\to 2\to 1$.

הבעיה ניתנת לפתרון בתכנון דינמי.
א. תארו נוסחה רקורסיבית לפתרון הבעיה.
ב. תארו את מבנה הנתונים התומך בפתרון הבעיה בתכנון דינמי. יש לתאר את סיבוכיות הזמן והמקום כפונקציה של n.
פתרון:
א. תכנון דינמי זה קשה לכן נתקוף את זה לאט. תמיד עלינו לבחור מבין שתי אפשרויות: לצמצם את הבעיה ל$n-1$, או לחלופין לבדוק האם קיימים שני מספרים כאלו. נטען טענה טריוואלית - תמיד קיימים שני מספרים כאלו אלא אם כן המספר ראשוני. לכן נתקוף את הבעיה כך: אם המספר ראשוני, נוריד באחד. בוודאות נקבל אגב מס' זוגי (אלא אם כן זה היה 2 שהוא הזוגי היחיד),ואז נוכל לתקוף את הבעיה עם מס' אחר. נתאר את נוסחת הנסיגה כדקלמן:

$$f(n) := \left\{ \begin{array}{cc} 0 & n=1 \\ min\{f(n-1)+1, f(\frac{n}{i})_{1\leq i\leq\sqrt{n}\wedge n\%i==0}\} & o.w \end{array} \right\}$$

מדוע הנוסחה נכונה? אם $ab=n$, אזי נרצה למצוא את האיברים שמקיימים זאת. נרצה את כל האיברים שיקיימו $a=\frac{n}{b}$ כלומר $n\%b==0$ וכן נראה כי נרוץ רק עד המספרים שיהיו שורש של n. מדוע?

$$b\geq a \Longrightarrow \frac{n}{a}\geq a \Longrightarrow n\geq a^2 \Longrightarrow \sqrt{n}\geq a$$

כלומר, נרצה לרוץ עד שורש אן, בשביל שהשוויון הנ"ל יתקיים תמיד.
ב. ראשית, ניצור מערך בגודל n שיכיל את הערכים $f(1), f(2), ...f(n)$. מערך כזה יהיה בגודל n. נראה כי עבור כל ערך, אנחנו נעבוד $\sqrt{i}$ עבודה על מנת לחשב את $f(\frac{n}{i})$ בכל אחד מהמשבצות. מכאן שהסיבוכיות הכוללת תהיה $\sum_{i=1}^{n}\sqrt{i}=n\sqrt{n}=n^{1.5}$. כלומר למלא את כל המערך יעלה לנו $O(n^{1.5})$. לבסוף, נקבל את הערך המינימלי עבור קלט n ב$f(n)$. סה"כ סיבוכיות מקום $O(n)$ וסיבוכיות זמן $O(n^{1.5})$.

1202 מועד ב' טופס שמואל קליין

שאלה 1

השאלה: מטפס הרים מתכנן מסלול מנקודת מוצא $S=A_0$ לנקודת יעד $T=A_{n+1}$. ועבור דרך של שרשרת בקתות הרים $A_1, ...A_n$, שכל אחת מהן יכולה לשמש לתחנת מנוחה לאורך המסלול. הבקתות מחוברות ע"י שבילים בעלי אורכים דומים, ולוקח יחידת זמן אחת להגיע מA_i לA_{i+1} לכל $1\leq i\leq n$. אך אם מטפס ההרים אינו עוצר בבקתה, זמן המעבר בשביל הבא מוכפל בקבוע $a>1$ (אם לא עצר המטייל, הוא התעייף..). הבקתות שונות זו מזו ולוקח זמן M_i בבקתה A_i כדי להתאושש ולנוח. בסה"כ יש מסלול - יש תחנות שבהם אנחנו עוברים. בכל בקתה אפשר לבחור לעצור לM_i זמן, ואם בוחרים שלא לעצור: בהתחלה זמן העצירה הוא 1, ואז בכל פעם הבאה זה יוכפל בa.
דוגמה: עבור $M_1, ...M_6(2.1, 0.6, 1.8, 0.9, 1.7, 1.3)$ ו$a=\frac{4}{3}$ אם מטפס ההרים מחליט לנוח רק בבקתות A_2 וA_4 אזי הזמן הכולל במסלול מA_0 לA_7יהיה

$$1+a+0.6+(1+a)+0.9+(1+a+a^2)=10.87$$

הסבר - מנקודת המוצא עד לנקודה A_1 לקח 1, הוא לא נח ולכן הוכפל בa ולקח עוד a זמן, אחכ עצר ולכן לקח $1+a$ כרגיל בין 2 ל3, הוא לא עצר בשלוש ולכן הזמן אחכ הוכפל שוב ואז לקח עוד $1+a+a^2$ וכו.
תאר אלגוריתם תכנון דינמי בשביל למצוא: א. את הזמן m הדרוש עבור המסלול המהיר ביותר מS לT. עשה זאת בשלושה שלבים: אלגוריתם נאיבי, רקורסיבי, ותכנון דינמי. ב. שחזור הפתרון - רשימת הבקתות בהם הוא צריך לנוח בהתאם לזמן האופטימלי.
פתרון: נתחיל מהאלגוריתם הנאיבי - בהינתן n בקתות ישנן 2 אפשרויות בכל פעם: לנוח בבקתה או שלא לנוח ולעדכן בהתאם. לפיכך, יש 2^n אפשרויות שונות. נצטרך לעבור על כולם ולחשב את הזמן ואז להחזר את המינימלי. זה יעלה $O(2^n)$.
נעבור לאלגוריתם הרקורסיבי - נגדיר $f(i)$ שתחזיר את הזמן המינימלי שלוקח לעבור מסלול של הבקתות $0, ...i$. כלומר, הזמן שיקח עד לעצירה בה. נגדיר את הנוסחה הבאה -

$$f(i) = \left\{ \begin{array}{cc} 0 & i=0 \\ min_{1\leq j\leq i}\{f(j)+\sum_{k=1}^{i-j}a^{k-1}+M_i\} & o.w \end{array} \right\}$$

הוכחת נכונות: נתחיל ממקרה הבסיס. וודאי שלהתחיל בראשון ולסיים יקח 0 כי עוד לא נכנסנו אל המסלול. אחרת, נרצה לקחת את הזמן המינימלי ביותר. נרוץ על האינדקסים $j \leq i$ ונבדוק מה הזמן המינימלי כאשר נבחר לעצור בj, ואז כל שאר הדרך מi עד לj לבחור להתקדם, ולכן נכפיל את המקדם שלנו, וכמובן נוסיף את הזמן שנחנו בבקתה הi, שכן עצרנו בה לנוח. הרקורסיה תמשיך וכך כל הפתרונות האפשריים ייספרו בתוך הרקורסיה.

נוכיח שהנוסחה אכן מחזירה את הפתרון המינימלי באינדוקציה על i.

בסיס: $i = 0$, אכן הפתרון המינימלי עבור מסלול עד 0 הוא 0 זמן כיוון שכלל עוד לא נכנסנו למסלול.

צעד ־ נניח שנקבל את המסלול הקצר ביותר לכל $k \leq n$. יהי המעבר ה$n+1$, אנחנו יודעים שהמסלול עד n הינו המסלול המינימלי שיכולנו לבחור. אנחנו רצים על כל האפשרויות עד אליו עם אינדקס j שלפי הנחת האינדוקציה נקבל את המינימלי וודאי, ונוסיף את $\sum a^k + M_{n+1}$ שיגידו כמה זמן צריך לנוח בתחנה ־ אכן הגיוני, וכן התוספת זמן שנצטרך להוסיף לאשר למכפיל שלנו. כל אחד מהמסלולים שנבחן יהיה בוודאות קטן מ$n+1$ ולכן יקיים את הנחת האינדוקציה ויחזיר את הזמן המינימלי, ולכן סה"כ יוחזר הזמן המינימלי ביותר.

פתרון תכנון דינמי: נשתמש במערך באורך $n+1$ ובו נאחסן את המידע $f(i)$. נקרא למערך A והוא יקיים לכל $0 \leq i \leq n$ כי $A[i] = f(i)$. נתחיל למלא את המערך, נעדכן $A[0] = f(0) = 0$ ואז נתחיל למלא משמאל לימין את המערך, נראה כי מילוי כל תא יעבור על j אפשרויות, במקרה הגרוע ביותר מדובר בn אפשרויות ולכן סה"כ מילוי כל תא יעלה לנו $O(n)$. ישנם n תאים ולכן סה"כ סיבוכיות זמן הריצה של הפתרון תעלה $O(n^2)$. כעת ־ הפתרון לבעיה הגדולה יהיה בתא $\}min\{f(i)$ שכן נצטרך לבדוק את כל האפשרויות וזה לעבור על n תאים לפי הגדרת הפונקציה שלנו, שם מוחזק הערך המינימלי ביותר של הבעיה. סיבוכיות זמן המקום תהיה $O(n)$. נשים לב כי איננו יכולים לשפר את סיבוכיות זמן המקום כיוון שבכל רגע נתון נזדקק לכל הנתונים הקודמים במערך, בחישוב המינימום. וכעת לשחזור הפתרון ־ נתחיל מהערך האחרון $A[n]$ ונבדוק את הזמן שלו. נשווה אותו לזמן של המסלול $A[n-1]$, וכך נדע האם בחרנו לנוח או שבחרנו להתקדם הלאה, בהתאם להפרש שהרי אם הוא 1 אזי נחנו ואם לא אזי בחרנו להתקדם הלאה ללא מנוחה, כך נמשיך הלאה אחורה ונוכל לשחזר את הפתרון שלנו ב$O(n)$. הערה ־ נשתמש במערך נוסף ובו נשמור לכל אינדקס באיזה תא לפניו הניב לנו את הפתרון, יענו איזה j הניב לנו תוצאה מינימלית. מש"ל.

שאלה 3

הבעיה: נאמר כי עץ T הינו סימטרי אם העץ סימטרי ביחס לשורש העץ. יהי T עץ בינארי. הנח שבכל צומת יש מצביע לבן ימני מצביע לבן שמאלי וכן ערך (לא רלוונטי לשאלה). הצע אלגוריתם לינארי שמקבל כקלט עץ בינארי ומחזיר אמת אם העץ סימטרי ושקר אחרת.

הפתרון: נרצה לפרק את הבעיה לתתי בעיות, נתחיל משורש העץ: אם הוא $null$ סיימנו ונחזיר אמת כי עץ ריק הוא בפרט סימטרי. כעת נעבור לבנים ־ אם איכשהו יש רק בן אחד ואין בן שני, נחזיר שקר שכן העץ לא סימטרי אם בצד אחד יש בן אחד ואין בן שסימטרי אליו. אחרת, נמשיך הלאה בירידה למטה בעץ: נרצה שאם נלך ימינה בעץ כלשהו, נלך שמאלה בשני (מפאת הסימטריות) מקרה קצה עיקרי הוא אם בה"כ הלכנו אכן ימינה בתת עץ ימני ושמאלה בשמאלי, קיבלנו מצב ששורש $null$ ובשני לא, אזי הסימטריות נשברה ונחזיר $false$. סה"כ נעבור על כל הקודקודים במקרה הגרוע (בו העץ אכן סימטרי ולא הייתה שבירה באמצע), ולכן סיבוכיות הזמן לינארית כלומר $O(n)$.

שאלה 4

הבעיה: יהי A מערך עם n איברים שונים. נאמר כי A הוא ממויין באופן מעגלי אם קיים k כך שאחרי הזזת האיברים k מקומות שמאלה באופן מעגלי (כלומר איברים שיוצאים מגבולות מערך שמאלי נכנסים בימני), המערך שיתקבל יהיה ממויין.

א. כתוב אלגוריתם שמקבל כקלט מערך A שכזה עם n איברים שונים ממויין באופן מעגלי ואינדקס $i \in \{1,...n\}$ ומחזיר 0 אם האיבר הגדול ביותר במערך נמצא משמאל לאינדקס i (כולל) כלומר ב$A\{1,..i\}$, אחרת יחזיר 1. זמן ריצה $O(1)$.

ב. כתוב אלגוריתם המקבל מערך עם n איברים שונים ממויין באופן מעגלי ומחזיר את מס' ההזזות המעגליות שיש לבצע על מנת שהמערך יהיה ממויין מהקטן לגדול. זמן ריצה $O(logn)$.

פתרון: א. האלגוריתם יעבוד כך ־ נחלק למקרים.

אם $i = 1$: נרצה לבדוק האם אנחנו במגמת עלייה. אם $A[1] < A[2]$ אזי תחזיר 1.

אם $i = n$: נשווה את $A[n]$ ל$A[1]$. אם $A[1] > A[n]$ אזי מצד שמאל של $i = n$ קיים האיבר הגדול ביותר ואיפשהו נמצאת הנקודה שצריך להזיז k פעמים.

אחרת: כלומר $1 < i < n$, נבצע לאיבר השוואה עם קודמיו. אם $A[i] > A[i-1]$ וגם $A[i] > A[i+1]$ אנחנו מצאנו את המקסימום, נחזיר 0. אחרת, השווה את $A[i]$ ל$A[n]$ ו$A[1]$. אם הוא גדול מ$A[1]$ וכן מ$A[n]$ השבירה מימין, אם גדול מ$A[1]$ וקטן מ$A[n]$ השבירה משמאל.

סה"כ, יש מס' סופי של פעולו־ת ובדיקות ולכן הפעולה מתבצעת ב$O(1)$

ב. כלומר עלינו למצוא את האינדקס k בו קורית השבירה. נפעל בהפרד ומשול, ונראה כי נגיע לנוסחת הנסיגה $T(n) = T(\frac{n}{2}) + O(1) = O(logn)$. נסתכל על האיבר האמצעי במערך, אם הוא קטן מ$A[1]$ וגם קטן מ$A[n]$ אזי שהשבירה

התרחשה בצד השמאלי, לכן נפעיל ברקורסיה על הצד השמאלי. אם האמצעי גדול מ$A[1]$ וכן גדול מ$A[n]$ אזי שהשבירה התרחשה בצד ימין של המערך ולכן נפנה לצד הימני. (שכן הוא גדול משניהם ובפרט גדול מ$A[1]$ לכן המערך משמאל ממויין והשבירה איפשהו בימין שכן האיבר האמצעי גדול מ$A[n]$). אחרת, מתי יודעים אם איבר הוא איבר השבירה ־ אם $A[i] > A[i-1]$ וגם $A[i] > A[i+1]$, במצב זה נחזיר את אינדקס שלו. סה"כ נוסחת הנסיגה מלאה נותנת את פתרון זמן הריצה הדרוש.

דרך שקולה ־ נשתמש בכל פעם באלגוריתם של סעיף א' על האיבר האמצעי ונרצה להבין היכן נמצא המקסימלי, בהכרח במקסימלי מתקיימת נק' השבירה.

מבחן אלגו' 1 0202 לדוגמה 2: שאלה 5

השאלה: בפרלמנט של מדינת אלגו־לנד ישנם n מושבים, הפרלמנט מורכב מk מפלגות שונות. נסמן את מס' המושבים בפרלמנט של המפלגה הi בs_i (כמובן $\sum_{i=1}^{k} s_i = n$) . קואליציה בפרלמנט חייבת להכיל יותר מחצי מהמושבים. והקואליציה מורכבת ממפלגות (כל חברי מפלגה מסויימת הולכים יחד לאופוזיציה או קואליציה) . לכל חבר כנסת בקואליציה משלמים משכורת גבוהה במיוחד ־ ולכן מטרת המדינה היא להרכיב קואליציה בגודל הקטן ביותר. (אך כמובן שחייבת לכלול לפחות חצי פרלמנט). בעיית קואליציית מינימום:

קלט: גדלי המפלגות $s_1, ...s_k$כך שגודל כל מפלגה מס' טבעי וסכומם שווה לn.

פלט: גודל קואליציה קטנה ביותר שאפשר להרכיב ממפלגות בגדלים אלו. תאר אלגוריתם תכנון דינמי שפותר את הבעיה. לפי השלבים ־ נאיבי, רקורסיבי, דינמי, שחזור פתרון.

פתרון: נתחיל מהפתרון הנאיבי ־ נרצה לעבור על כל סוגי ההרכבים השונים, ולקחת את ההרכב המינימלי שסכומו גדול מ$\frac{n}{2}$. נראה כי בהינתן k מפלגות מס' האפשרויות יהיה אקספוננציאלי ־ לכן ננטוש את הפתרון מראש. שכן לכל מפלגה אפשרות להכנס או לא, שזה 2^k אפשרויות ואז צריך לחפש את האפשרויות בהם הגודל גדול שווה מ$\frac{n}{2}$ וכן המינימלי. אקספוננציאלי.

פתרון רקורסיבי: לכל מפלגה יש שתי אפשרויות, להכנס לקואליציה או שלא. כמו כן נרצה להתייחס לתנאי של הסף. לכן אנחנו נגדיר את הפונקציה הבאה: $f(i,x)$ כאשר x יהיה הסף הדרוש להקמת קואליציה, ואנחנו בכל פעם נוריד ממנו את הערך שקיבלנו. פתרון טוב יתקבל כאשר $x < 0$ שכן נאתחל את x להיות $\frac{n}{2}$ (בערך עליון). הפונקציה תחזיר את מס' חברי הפרלמנט שירכיבו את הקואליציה כאשר היא מורכבת מהמפלגות $1, ...i$ כאשר $1 \le i \le k$

נגדיר את הנוסחה הרקורסיבית הבאה:

$$f(i,x) := \left\{ \begin{array}{cc} 0 & i > k \wedge x \le 0 \\ \infty & i > k \wedge x > 0 \\ min\{f(i+1, x-s_i) + s_i, f(i+1,x)\} & o.w \end{array} \right\}$$

הוכחת נכונות: הכניסה לנוסחה מתחילה כאשר $i = 0$, אנחנו הולכים לרוץ ולהתייחס לשני מקרי קצה: אם ערך הi עבר את k כלומר $i > k$ אזי נחלק למקרים: עברנו על כל המפלגות ולכן יש שני אפשרויות. אם הx קטן מאפס, אזי עברנו את כל חברי הקואליציה שנדרשים לנו ולכן נסיים ולא נוסיף עוד. אם הx עדיין גדול מאפס אזי לא עברנו את הרף המינימלי להקמת קואליציה ולכן אנחנו נחזיר אנסוף שיסמן לנו שלא ניתן להרכיב קואליציה. אחרת, אנחנו במצב שבו נהיה רוב הזמן: נלך על הבחירה המינימלית משני האפשרויות ־ לקחת או לא לקחת. אם נקח אזי נתקדם הלאה כי אי אפשר לקחת מפלגה פעמיים וכן נוריד את ערך s_j מx , אם לא נקח רק נתקדם הלאה.

תכנון דינמי: נרצה ליצור מטריצה בגודל $k+1 \times n+1$. מדוע? כיוון שיש n ערכים אפשריים להרכבת קואליציה. נמלא לפי התנאי בסיס שמפורטים כעת, כאשר אם מתקיים במהלך המילוי כי $i > k$ וכן $x \le 0$ אז נשים אפס כי לא צריך עוד חברי קואליציה ואם $x > 0$ נשים אנסוף כי לא נגיע לפתרון, אחרת נפעל לפי האפשרות השלישית. נמלא לפי השורות בשל התלות בנוסחה. נראה כי הפתרון לבעיה יכול להמצא בכל אחד מהתאים המקיימים את התנאים. ולכן אנחנו נאלץ לסרוק את המטריצה פעם נוספת לבסוף למציאת הפתרון הסופי. נראה כי עלות הזמן והמיקום היא $O(nk)$ וכן לא ניתן לשפר יעילות כיוון שזקוקים בכל רגע נתון לכל תאי המטריצה.

מבחן לדוגמה 3 אלגו' 1 שאלה 5

השאלה: בממלכת הנמלים העולם כולו מתנהל באופן חד מימדי. על הישר הממשי. ישנן n נמלים ולכל נמלה יש בית שהוא נקודה על הישר הממשי (מס' ממשי כלשהו). נניח שאין יותר מבית אחד בכל נקודה. ועדת הבחירות בממלכת הנמלים מעוניינת להערך לבחירות הקרובות ולכן הוועדה נערכת להקמת קלפיות. לצורך כך צריכה הוועדה לבחור k מספרים (עבור k נתון) מתוך המספרים בהם גרות הנמלים והוועדה תקים קלפיות בk המספרים הללו. באם הקלפיות יוקמו בקבוצה C (שהיא תת קבוצה של המספרים בהם יש בתים לנמלים) , כאשר נמלה שגרה במס' a תלך להצביע אזי נמלה זו תלך להצביע בקלפי הקרוב לביתה. שמרחקו הוא כמובן $min_{c \in C}|a-c|$. העלות(מרחק ההליכה) המתקבלת מהקמת קלפיות בקבוצה C מסויימת היא $cost(c) = max_{a \in A}\{min_{c \in C}|a-c|\}$. מטרתכם היא לסייע לוועדת הבחירות

לבחור את C כך שהמרחק הגדול ביותר מבית כלשהו לקלפי הקרובה יהיה המינימלי ככל שניתן (כלומר $cost(c)$ יהיה מינימלי). שימו לב ־ אין באפשרות ועדת הקלפי להקים קלפיות במספר בו לא מתגוררת אף נמלה. (הערה ־ הקלפי ממש קם בבית כלשהו של אחת הנמלות).

בעיית מציאת k־קלפיות

קלט: קבוצה A של n מספרים שונים ומס' טבעי $k \leq n$

פלט: תת קבוצה $C \subseteq A$ בגודל $|C| = k$ כך שמביאה למינימום את $cost(c) = max_{a \in A}\{min_{c \in C}|a - c|\}$.

חשוב: בשאלה זו נסתפק במציאת המרחק הגדול ביותר שנמלה תצטרך ללכת בפתרון האופטימלי שמביא את $cost(c)$ למינימום, ולא את הקבוצה C ־ כלומר אין צורך בשחזור הפתרון. תאר אלגוריתם נאיבי, לאחר מכן רקורסיבי עם נוסחת נסיגה מתמטית, לאחר מכן נתח אותו לפתרון תכנון דינמי. בהצלחה!

פתרון: נתחיל מהפתרון הנאיבי ־ בהינתן n אפשרויות להקמת קלפיות וכן k קלפיות שצריך לבחור בהם, לפי בדידה זו בחירה ללא חזרה וללא חשיבות ולכן $\binom{n}{k}$־ פתרון אקספוננציאלי שכן

$$\binom{n}{k} = \frac{n!}{(n-k)!k!} = O(\frac{n!}{(n-k)!k!}) = O(n^k)$$

לאחר מכן נרצה לסרוק את כל האפשרויות הללו, ולחשב את המרחק של כל איבר בקבוצה מקלפי, סה"כ נקבל ממילא $\Omega(\binom{n}{k})$ שזה אקספוננציאלי ולא בא בחשבון.

נעבור לפתרון הרקורסיבי ־ שנוכל לתאר את נוסחת הנסיגה הבאה. נגדיר $f(j, i)$ כמינימום $cost$ כאשר מציבים i קלפיות עבור הנמלים $\{1, ...j\}$. כאשר $1 \leq i \leq k$ וכן $1 \leq j \leq n$. נראה כי תמיד $j \geq i$ שכן מס' הנמלים יהיה חייב להיות גדול שווה ממס' הקלפיות שכן הקלפיות קמות על בתי הנמלים. הבעיה ביקשה שנמצא את המרחק הגדול ביותר בפתרון האופטימלי. לכן אנחנו נרצה את המזעור של $cost(c)$, והוא המרחק הגדול ביותר שנמלה הולכת. נראה כי לנמלה יש שני אפשרויות: אם יציבו קלפי בביתה והיא לא תצטרך ללכת, ואם לא מציבים קלפי בביתה.

**הערה כי זה מבלבל ־ אנחנו רוצים את המרחק המינימלי, זה מה שמעניין. אחכ בהינתן אותו מרחק, זה יהיה גם המקסימלי מתוך הקבוצה שלנו אבל הרעיון הוא למצוא מרחק מינימלי (הכי גדול בקבוצה).

כעת, נניח כי הקלפי הכי ימנית תקרא t. אזי התחלקנו לשני חלקים, נמלים $i - t$ ילכו אל t והנמלים $t + 1 - j$ לא מכוסות ע"י שום קלפי וצריך לפתור להם גם. נסמן a_L כמיקום נמלה l.

$$f(j,i) := \left\{ \begin{array}{cc} \infty & i > j \\ \infty & i = 0 || (i > 0 \wedge j = 0) \\ 0 & j = 0 \\ min_{i \leq t \leq j}\{max\{f(t, i-1), max_{t \leq l \leq j}(a_l - a_t)\} & o.w \end{array} \right\}$$

הוכחת נכונות: אם מס' הקלפיות גדול ממס' נמלים או לחלופין מס' קלפיות הוא אפס, או שהוא גדול מאפס אבל יש אפס נמלים ־ נחזיר אנסוף. כל אלו לא רלוונטים לנו לשאלה. אחרת, אם אין נמלות כלל אבל אין קלפיות, נחזיר 0. כעת נעבור למקרה האחרון ־ אנחנו נרצה לעבור על כל הt כך שיהיה גדול שווה ממס' הקלפיות אך קטן שווה ממס' הנמלות, ואז אנחנו נחשב את המקסימום בין שני אפשרויות: t הינו הקטע בו "חתכנו" את המפה עם הקלפי הימנית ביותר. במצב הראשון זה נפל אצל הנמלה בבית הקלפי ־ ואז הבעיה הופכת להיות עם מס' נמלים t שנותרו וכן $i - 1$ קלפיות כי הורדנו קלפי בבית של t. או לחלופין, נחשב את המקסימום של הפרש המיקומים כאשר נעזר בפרמטר l שירוץ בין t לj (שאר הקטע שלנו) ונחשב את המרחק המקסימלי.

תכנון דינמי: ניצור טבלה בגודל $n+1 \times k+1$ ונמלא אותה לפי נוסחת הנסיגה מעלה. כיצד? לפי עמודות (מהנוסחה ניתן לראות). הפתרון לבעיה יהיה ב$A[n, k]$. באשר לסיבוכיות זמן הריצה, נראה כי נעבור $j - i$ פעמים (מעבר על t של בחירת מקסימום בין שני ביטווים, ושם נחשב שוב $j - t$ חישובים כאשר נרוץ על l, סה"כ מילוי כל תא יקח n^2 זמן ויש n^2 תאים ־ לכן סיבוכיות זמן הריצה של הבעיה יהיה $O(n^3k)$ וכן סיבוכיות מקום של $O(nk)$.

אלגו' 1 מבחן 4102 מועד א שאלה 1

הבעיה: סדרת מספרים $w[1], ...w[k]$ תקרא מתחלפת אם לכל i זוגי $w[i] > w[i+1]$ ולכל i אי זוגי $w[i] < w[i+1]$. בעיית תת הסדרה המתחלפת ־

קלט: סדרת מספרים A

פלט: אורך תת הסדרה המתחלפת הארוכה ביותר.

כתוב פתרון תכנון דינמי לפי השלבים הבאים: נאיבי, רקורסיבי (נוסחת נסיגה מתמטית), תכנון דינמי וניתוח סיבוכיות זמן ומקום.

פתרון: נתחיל מהפתרון הנאיבי. בהינתן n איברים, יש 2^n תתי סדרות שלהם. לפיכך, נוכל לעבור על כל אחת מתתי הסדרות. מעבר על כל אחת הוא $O(n)$. נבדוק לאורך הבדיקה האם היא אכן מתחלפת ע"י שמירת אינדקסים

והשוואות. ובכך נשמור את האורך הגדול ביותר של סדרה כזו ונעדכן. בסוף נחזיר את האורך הכי גדול שמצאנו. סה"כ סיבוכיות זמן ריצה של $O(n2^n)$. די גרוע.

נעבור לפתרון הרקורסיבי. נרצה לשמור שלושה אינדקסים.$f(i,j,k)$ כפונקציה שתחזיר את אורך תת הסדרה המתחלפת עבור סדרה שהחלה מאינדקס i וכעת נמצאת באינדקס j בכיוון k.

כאשרi האינדקס הקודם שביקרנו בו jהאינדקס בו אנחנו נמצאים כרגע ־ k = הכיוון (אם $k = 1$ נרצה עלייה, $k = 0$ אם נרצה ירידה)

כעת נראה כי ישנם שני אפשרויות: אם $k = 1$ כלומר אנחנו בעליה, אם $A[j] > A[i]$ נוסיף אחד ונעבור לתת הבעיה של $j + 1$. אם $k = 0$ אזי אם $A[j] < A[i]$ נוסיף אחד ונעבור לתת הבעיה של $j + 1$. כמו כן בכל אפשרות נוכל שלא לבחור באיבר ולהתקדם הלאה. נראה כי זו גישת הלקחת או לא לקחת. אם $k = 1$ וכן $A[j] > A[i]$ או השבירה השנייה שיכולה להתרחש, נחזיר במקרה כזה $-\infty$ שכן לא ניתן להמשיך את תת הסדרה כי היא נשברה.

$$f(i,j,k) := \left\{ \begin{array}{cc} 0 & j > n \\ -\infty & k = 1 \wedge A[j] < A[i] \\ max\{1 + f(j, j+1, 0), f(i, j+1, k)\} & k = 1 \wedge A[j] > A[i] \\ max\{1 + f(j, j+1, 1), f(i, j+1, k)\} & k = 0 \wedge A[j] < A[i] \\ -\infty & k = 0 \wedge A[j] > A[i] \end{array} \right\}$$

נעבור לפתרון התכנון הדינמי: נגדיר מערך תלת מימדי בגודל $n + 1 \times n + 1 \times 2$ כאשר n הוא אורך המערך. לפי הנוסחה, עמודה ראשונה של 2 יתאר את הכיוון, אחריה זה j שיתאר מיקום נוכחי וכן i מיקום שהחלה תת הסדרה. במקומות ש$j > n$ נמלא אפס, וכן אם $k = 1 \wedge A[j] < A[i]$ או $k = 0 \wedge A[j] > A[i]$ נמלא מינוס אנסוף. מילוי כל תא יקח $O(1)$. כעת, נראה כי כדאי לנו למלא את המטריצה התלת מימדית הזו במקביל כאשר נרוץ על k ועל j, מה שיבטיח סיבוכיות של $O(1)$ למילוי תא שכן בוחרים מקסימום בין שני תאים שכבר קיימים. לבסוף, אנחנו נרצה לחפש את המחרוזת הכי ארוכה באורך n ולכן הפתרון לבעיה יהיה $max\{[1, n, 1], [1, n, 0]\}$. סיבוכיות הזמן ־ מילוי של n^2 תאים שכל אחד $O(1)$ ולכן $O(n^2)$. סיבוכיות מקום ־ $O(n + 1 * n + 1 * 2) = O(n^2)$. ניתן לצמצם מקום ולהחליט ששומרים בכל פעם שני עמודות j קודמות ולהוריד את המקום ל$O(n)$.

מבחן מבנה נתונים 2202 קיץ מועד א'

שאלה 1

נתונה מטריצה (מערך דו ממדי) משולשית תחתונה ־ כלומר כל האיברים מעל האלכסון הראשי הינם אפסים, נסמנה A. מסלול על המערך A מתחיל במיקום $[1, 1]$ (אין אינדקס אפס אצלנו), ומסתיים באחד המיקומים בשורה התחתונה. צעד במסלול ממשבצת $[i, j]$ יכול להיות ל$[i + 1, j]$ (למטה באותה עמודה) או $[i + 1, j + 1]$ (באלכסון מטה). ערך מסלול הוא סכום הערכים הנמצאים על המסלול. המטרה היא מציאת המסלול שערכו הוא הגדול ביותר. הבעיה ניתנת לפתרון בעזרת תכנון דינמי.

א. הציגו פתרון נאיבי ופסלו אותו. אחר כך הציגו נוסחה רקורסיבית מתמטית לפתרון הבעיה. יש לפרט ולהוכיח נכונות (לא פורמלי).

ב. הפוך נוסחה רקורסיבית לפתרון תכנון דינמי, וקבע סיבוכיות זמן ומקום. האם ניתן לצמצם שימוש בזכרון?

ג. איך ניתן לשחזר את הפתרון?

פתרון:

נתחיל מהפתרון הנאיבי, מס' האפשרויות להגיע מנקודה לנקודה הוא כאשר אנו בוחרים בכל שלב מבין שני אפשרויות ואז צריך לחפש את הערך המקסימלי ־ $O(2^{n-1})$. אחכ נרצה גם לחשב בכל מסלול את הערך ולכן סה"כ $O(n2^{n-1})$. אקספוננציאלי ולא בא בחשבון.

נעבור לפתרון הרקורסיבי: נשים לב שבכל רגע נתון המשתמש חייב לבצע צעד אחד, לכן בפניו שני אפשרויות. נגדיר $f(i, j)$ שתחזיר את הערך המקסימלי למסלול שהחל בנקודה $[1, 1]$ והסתיים בנקודה $[i, j]$. כעת ניגש לנוסחת הנסיגה הבאה ־

$$f(i,j) = \left\{ \begin{array}{cc} A[1,1] & i = j = 1 \\ -\infty & i > n || j > n, j > i \\ A[i,j] & i = n \\ max\{A[i,j] + f(i-1, j), A[i,j] + f(i-1, j-1) & o.w \end{array} \right\}$$

הוכחת נכונות והסבר הנוסחה: אם אנחנו הגענו נמצאים בתא הראשון , אזי פשוט נחזיר את ערכו שכן יש אפשרות אחת. אם $i > n$ או $j > n$ משמע ניסינו לחרוג מגבולות המטריצה ־ אנחנו לא חובבי $exceptions$ ולכן מיד נעצור את המשתמש עם $-\infty$ על מנת שפתרון זה יפסל (שהרי בוחרים מקסימום), אם $i = n$ משמע אנחנו על השורה התחתונה

ויש להפסיק את המסלול. אחרת, אנחנו נבחר בכל פעם את המקסימום מבין שני אפשרויות שנותנות לנו, כאשר נקח את הערך הנוכחי ועוד אפשרות שללכת לאלכסון מטה או ללכת מטה בעמודה. מדוע זה יתן את הפתרון המקסימלי? נוכיח באינדוקציה את הטענה

טענה: נוסחת הנסיגה תניב את הפתרון המקסימלי לבעיה.

הוכחה: באינדוקציה. בסיס - $n=1$, לפי הנוסחה נקבל את $A[1,1]$ שכן הוא הפתרון המקסימלי לתת הבעיה שמתחילה ב$A[1,1]$ וגם מסתיימת בו. צעד - נניח כי הטענה נכונה לכל מסלול שקטן מn צעדים. יהי מסלול בעל n צעדים שהסתיים בנקודה $[x_i, x_j]$. לפי הגדרת השאלה - הגענו אליו משני אפשרויות: או $x_{i-1,j-1}$ (מלמעלה באלכסון). או $x_{i-1,j}$. שני האפשרויות הם מסלולים שאורכם קטן מn. לפיכך, הם מקיימים את הנחת האינדוקציה ומחזירים את הערך המקסימלי. לכן, אנחנו יודעים שכל אחד מהם מחזיר ערך מקסימלי נכון לשעתו, ולכן אנחנו רוצים לקחת את הערך של כל אחד מתתי המסלולים האלו כלומר $f(x_{i-1}x_j)$ וכן $f(x_{i-1}x_{j-1})$ להוסיף להם את הערך $A[x_i, x_j]$ ולקבל את המקסימום. נראה כי כיוון שכל אחד מהם הוא הפתרון האופטמלי למסלול, הוספת ערך קבוע לא משנה את הערך המקסימלי שיכול להתקבל (שכן גם אם מדובר במס' שלילי, עדיין היינו חייבים לעבור במסלול זה להגעה לשורה האחרונה) ולכן הפתרון המקסימלי יישמר. סה"כ נקבל פתרון אופטימלי מקסימלי לבעיה.

כעת נעבור לתכנון הדינמי - נרצה לשמור מטריצה בגודל $n \times n$ (אין ערך אפס) ונסמנה B. בכל תא במטריצה נשמור את $f(i,j)$. ראשית, נתחיל למלא בכל החלקים שמעל האלכסון הראשי ב$-\infty$. אמנם אין דרך להתקרב לשם אבל בשביל הנוחות - אח"כ נתחיל לפעול לפי נוסחת הנסיגה: נראה כי נראה למלא במילוי בצורה שונה מבדרך כלל - אנחנו נתחיל למלא מהסוף להתחלה, את השורה האחרונה נראה שהיא מתאימה למקרה $i=n$ ולכן את כל השורה האחרונה של המטריצה נמלא בערכי $B[i,j] = A[i,j]$. אח"כ נעבור לשורה שלפניה ונמלא אותה לפי $o.w$ בהתאם. נראה כי כל תא מבצע מס' פעולות סופיות ולכן מילוי כל תא יקח $O(1)$, ישנם n^2 תאים ולכן יעלה לנו בסה"כ $O(n^2)$ גם בזמן וגם במקום למלא את המטריצה. נראה כי הערך בו אנחנו מעוניינים נמצא בשורה האחרונה היכן שהוא ולכן נסרוק שם למציאת הפתרון המקסימלי. נעיר כי עם שמירת אינדקסים נוכל לצמצם ולשמור רק שתי שורות (נוכחית וקודמת) ולצמצם את השימוש בזכרון ל$O(n)$. כעת לשחזור הפתרון - נתחיל בפתרון המקסימלי בשורה האחרונה, ונשאל את עצמנו בכל שלב - מי יכל להביא אותי לכאן? או שזה שנמצא איתי באלכסון, או זה שמעליי. נחשב את הערך שלהם בתוספת הערך $A[i,j]$ ונראה מי יכל להביא אותי למקום הזה, כך נמשיך עד שנגיע ל$A[1,1]$ לבסוף ושחזרנו את הפתרון. כמה זה יעלה? $O(n)$ זמן שכן אנחנו מבצעים עד n שאילתות כאלו. מש"ל.

שאלה 2

מבנה נתונים "ערימת חציון" תומך בפעולות הבאות: אתחול המבנה על n נתונים, הוצאת החציון מהמבנה, החזרת ערך החציון (ללא שינוי המבנה). הוכיחו/ הפריכו את הטענה הבאה: ניתן לממש מבנה זה בזמני הריצה הבאים לכל פעולה - אתחול מבנה ב$O(n)$, הוצאת חציון ב$o(logn)$ והחזרת ערך החציון ב$O(1)$.

פתרון: בדרך כלל שאלות כאלו הינם שאלות הפרכה שמבקשות להניח בשלילה שקיים מבנה, ואז לבצע מיון מבוסס השוואות ב$o(nlogn)$ בסתירה לחסם תחתון $\Omega(nlogn)$. נקווה שיעבוד לנו. נב"ש שקיים מבנה כזה בזמני הריצה הנתונים. ניצור מערך בגודל n. נעתיק אליו את החציון. נוציא את החציון מהמבנה המקורי. עלה לנו עד כה $o(logn)$ וכן אתחול המבנה עצמו ב$O(n)$. כעת נחזור על הפעולה הבאה - ניקח את החציון שבראש המבנה החדש עם $n-1$ איברים, נוציא אותו מהמבנה ב$o(logn)$ ונשאל האם הוא גדול הוא קטן מאמצע המערך. נחזיק שני פוינטרים שיצביעו עליו. כעת נמשיך כך עד להוצאת כל האיברים מהמבנה - נחזיר את ערך החציון ונבדוק האם הוא קטן או גדול מהאמצע, אם הוא קטן נכניס אותו צמוד לפוינטר משמאל, ואם הוא גדול מהחציון נכניס אותו צמוד לפוינטר מימין במערך, ונעדכן פוינטרים בהתאם (אם הוא היה נכנס משמאל היינו לוקחים פוינטר אחד מהאמצעי שמאלה ואם היה נכנס מימין היינו לוקחים פוינטר אחד מהאמצעי ימינה את השני). סה"כ נקבל מיון מבוסס השוואות שמיין את המערך והתבצע ב$o(nlogn)$, בסתירה. נכונות: לאחר שהוצאנו את איבר החציון מהמבנה, החציון הבא הוא האיבר הקטן ביותר מהאיברים הגדולים שמעבר לחציונים שהוצאנו או לחלופין האיבר הגדול ביותר מבין האיברים שמשמאל לאיברים שמיינו.

שאלה 3

בהינתן a,b,c נאמר כי b קרוב יותר לa מאשר c קרוב לa אם $|a-b|<|a-c|$. כתוב אלגוריתם יעיל ככל הניתן המקבל מערך A של מס' ממשים ומשתנה k (טבעי) וx ממשי ומחזיר את k האיברים במערך A הקרובים ביותר למשתנה x.

פתרון: נאיבי - נחשב לכל אחד מהאיברים את המרחק שלהם מהמשתנה x, זה יעלה $O(n)$. נשמור זאת ב"מערך המרחקים". נרצה ליצור ממנו ערימת מינימום - זה יעלה $O(n)$. לאחר מכן נרצה להוציא את k המרחקים הקטנים ביותר, זה יעלה $klogn$. סה"כ הנאיבי הוא $O(n+klogn) < O(nlogn)$

ננסה לייעל - שהרי השאלה לא יכולה להיות פשוטה כל כך. שוב נרצה לחשב לכל איבר בקבוצה את המרחק שלהם מx. נשמור זאת במרחק המרחקים עם פוינטר שיצביע לאיבר שמתאים למרחק (שהרי לא נוכל אח"כ לשחזר את המרחק שהרי המרחק יכול להיות חיובי או שלילי, איננו יודעים אם התבצע $x-a$ או $a-x$. בכל מקרה, זה יעלה $O(n)$. כעת נפעיל אלגוריתם סלקט למצוא את האיבר הk בגודלו, זה יעלה לנו $O(n)$, כעת נרצה לעבור איבר אבר

במערך המרחקים ולהשוות אותו לאיבר הk, אם הוא קטן ממנו מעולה זה איבר שנרצה להחזיר ואם גדול ממנו אז כבר לא. סריקה שכזו תעלה גם $O(n)$. סה"כ פתרון ב$O(n)$ - לא ניתן יעיל מזה כי חייבים לעבור על כל האיברים.

שאלה 4 - אינדוקציה על עצים

יהי T עץ AVL שלם (מלא - לכל אחד יש שניים, והעלים באותו עומק) בגובה $h \geq 0$. ותהי $k_1 < ..... < k_{2^{h+1}}$ סדרה עולה של 2^{h+1} ערכים כך שערכו של k_1 גדול מכל מפתח שנמצא בעץ T (ולכן גם שאר הערכים). הוכיחו באינדוקציה על מבנה העץ כי לאחר הוספת הסדרה של 2^{h+1} המפתחות, כלומר הכנסת k_1 ואחכ k_2 וכו' , נקבל עץ שלם בגובה $h+1$. 52 נקודות.

פתרון: נוכיח באינדוקציה על גובה העץ h.

בסיס - $h = 0$, אם גובה העץ הוא 0 אזי מדובר באיבר אחד בודד. כעת נרצה להוסיף לעץ שני ערכים ($2^{0+1} = 2$) שגדולים מהערך שנסמנו x שנמצא בשורש העץ שלנו. ראשית נכניס את הערך הראשון x_1, נכניסו מימין כיוון שהוא גדול מx, כעת נרצה להכניס את x_2, נכניסו מתחת לx_1ואז הפרש גבהים -2 אז צריך רוטציה, לאחר הרוטציה נקבל כי בשורש x_1, מימין x_2 ומשמאל x, סה"כ עץ בינארי שלם (כל העלים באותו עומק) וכן מלא כי יש שני עלים. והגובה הוא $1 + 0 = 1$. כנדרש.

צעד - נניח נכונות לכל עץ שלם ומעבר כזה בגובה $k < h$. יהי עץ T שלם מגובה h. נרצה לבצע את ההכנסה של כל האיברים $k_1, ...k_{2^{h+1}}$. את העץ נחלק לשלושה: השורש שיסומן x, העץ הימני T_R והעץ השמאלי T_L. גובה כל אחד מהעצים הקטנים הוא $h - 1$. נסתכל על כל המפתחות עד k_{2^h}. נכניסם לעץ T_R. לפי הנחת האינדוקציה, כיוון שהכנסנו $2^{h-1+1} = 2^h$ מפתחות, נשמר העץ השלם, והוא בגובה $h - 1 + 1 = h$. כלומר עד כה הכנסנו לעץ 2^h ערכים חדשים, קיבלנו שורש של x, עץ ימני בגובה h ועץ שמאלי בגובה $h - 1$. לפי חוקי AVL -חוקי. הפרש הגבהים הוא -1 שזה חוקי. כעת נרצה להכניס את האיברים $2^h + 1, ...2^{h+1}$, יש 2^h מפתחות כאלו. כאשר נוסיף את הראשון $2^h + 1$, נקבל רוטציה פשוטה בה הוא נכניס מימין למטה (הכי גדול), השורש של T_Rיהפוך לשורש החדש והשורש הישן יהיה הבן השמאלי. כך נמשיך עבור כל הערכים $2^h + 2, ...2^{h+1}$ונכניסם, נקבל שהעץ המקורי היה משמאל בגובה $h - 1$ והפך לh, גם העץ מימין בגובה h והשורש הינו T_R. קיבלנו עץ שלם, כנדרש.

אלגוריתמים 1 מבחן 4202 סמסטר ב' שאלה 1

השאלה: נתונה מחרוזת $S = a_1, a_2, ...a_n$ (מערך מספרים בגודל n) כאשר התווים של S הם מספרים שלמים. המטרה היא לחשב את תת הסדרה היורדת הרציפה הארוכה ביותר. תת סדרה כזו מוגדרת כך: $a_{k1}, a_{k2}, ...a_{kw}$ כך שלכל $j = i + 1$ וכן $i \geq 1$ מתקיים $k_j - k_i = 1$ וכן $a_{ki} > a_{kj}$. כתוב אלגוריתם תכנון דינמי הכולל פתרון נאיבי, נוסחה רקורסיבית מתמטית, תכנון דינמי שמחזיר את אורך תת הסדרה הארוכה ביותר (לא חשובה הסדרה עצמה). נתח סיבוכיות זמן ריצה ומקום. האם ניתן לחסוך במקום.

פתרון: נתחיל בנאיבי. נרצה לדעת כמה תתי סדרות רציפות יש. בכללי יש 2^n תתי סדרות אבל זה כולל גם כאלו רציפות. נשאל כמה אפשרויות יש לתתי סדרות רציפות. נראה כי יש n אפשרויות לאינדקס ההתחלה, וכן אם i אינדקס ההתחל לאינדקס הסוף יש $i, i + 1, ...n$ אפשרויות. לפיכך, לכל i ישנם $n - i + 1$ אפשרויות עבור j. ולכן

$$\sum_{i=1}^{n}(n - i + 1) = \frac{n(n + 1)}{2} = O(n^2)$$

לכל תת סדרה נרצה לבדוק האם היא עולה או יורדת, נזדקק לעבור עליה ב$O(n)$. מכאן שסה"כ הפתרון הנאיבי הוא ב$O(n^3)$זמן.

נעבור לרקורסיבי. כעת נרצה להגדיר פונקציה $f(i)$ שתתאר את גודל תת הסדרה היורדת הרציפה המקסימלי שנגמר באינדקס i. מכאן נוכל להגדיר את נוסחת הנסיגה הבאה:

$$f(i) := \left\{ \begin{array}{cc} 1 & o.w \\ f(i-1) + 1 & A[i] < A[i-1] \end{array} \right\}$$

הסבר: $i = 1$, במקרה הזה תת הסדרה המקסימלית שמסתיימת באינדקס 1 הייתה חייבת להתחיל בו, סדרה בעלת איבר אחד אכן יורדת באופן ריק ולכן במצב זה יוחזר הערך המקסימלי - 1.

אחרת, אם אכן מתקיים כי הסדרה יורדת נחשב את הערך הקודם ונוסיף אחד לאורך וכך נמשיך

נשמור את הנתונים במערך בגודל n, נסרוק אותו והפתרון יהיה $max_{1 \leq i \leq n}\{f(i)\}$ שכן תת סדרה מקסימלית יכולה להיות בכל ערך. אופן המילוי יתבצע משמאל לימין כמובן במערך. סה"כ זמן הריצה יהיה $n + n = O(n)$ וכן סיבוכיות המקום $O(n)$. נראה כי הפתרון יהיה ב$B[n]$ שם יש את אורך תת הסדרה המקסימלית עד לאורך n, כעת נוכל לחסוך במקום אם נשמור בכל פעם את האיבר הקודם, ונוכל להגיע לסיבוכיות זמן של $O(n)$ וריצה $O(1)$.

מבחן 3202 סמסטר א'

שאלה 1: ראינו בהרצאה כיצד להכפיל 2 מספרים X וY כל אחד בעל n סיביות בשיטת הפרד ומשול ב$O(n^2)$ כאשר כל מס' חולק לשניים באורך $\frac{n}{2}$. כאשר צמצמנו את מס' הקריאות הרקורסיביות מ4 ל3 הסיבוכיות ירדה ל$O(n^{1.58})$. שאלה זו תדון בהצעה לחלק כל מספר ל3 חלקים באורך שווה ולא ל2.

א. הגדירו את X וY כפונקציה של שלושת חלקיהם X_1, X_2, X_3וY_1, Y_2, Y_3

ב. הראו כיצד ניתן לחשב את XY ע"י 9 קריאות רקורסיביות (הפרד) ועוד עבודת איסוף התוצאות שעלותה לינארית בn (ומשול)

ג. מה סיבוכיות השיטה?

ד. בדומה למה שראינו במקרה חלוקה ל2, הראו כיצד בעזרת חישוב הערכים הבאים: $E = (X_1 + X_2)(Y_1 + Y_2)$ $F = (X_2 + X_3)(Y_2 + Y_3)$ ו$G = (X_1 + X_3)(Y_1 + Y_3)$ ניתן להוריד את מס' הקריאות רהקורסיביות מ9 ל6.

ה. מה הנוסחה המתקבלת עבור הסיבוכיות?

בונוס - 3 נקודות, האם כדאי לחלק ל3?

פתרון: כפי שראינו בהרצאה נגדיר הפעם לשלושה חלקים

$$X = X_1 * 2^{\frac{2n}{3}} + X_2 * 2^{\frac{n}{3}} + X_3$$

$$Y = Y_1 * 2^{\frac{2n}{3}} + Y_2 * 2^{\frac{n}{3}} + Y_3$$

נראה כעת כי

$$XY = (X_1 * 2^{\frac{2n}{3}} + X_2 * 2^{\frac{n}{3}} + X_3)(Y_1 * 2^{\frac{2n}{3}} + Y_2 * 2^{\frac{n}{3}} + Y_3) =$$

$$X_1Y_1 * 2^{\frac{4}{3}n} + X_1Y_2 * 2^n + X_1Y_3 2^{\frac{2n}{3}} + X_2Y_1 2^n + X_2Y_2 2^{\frac{2n}{3}} + X_2Y_3 2^{\frac{n}{3}} + X_3Y_1 2^{\frac{2n}{3}} + X_3Y_2 2^{\frac{n}{3}} + X_3Y_3$$

סה"כ קיבלנו 9 קריאות, איסוף תוצאות לינארי ולכן

$$T(n) = 9T(\frac{n}{3}) + O(n)$$

לפי מאסטר, $n^{log_3 9} = n^2$. כלומר הסיבוכיות הינה $\Theta(n^2)$.

נראה כי -

$$XY = 2^{\frac{4n}{3}}(X_1Y_1) + 2^n(X_1Y_2 + X_2Y_1) + 2^{\frac{2n}{3}}(X_1Y_3 + X_2Y_2 + X_3Y_1) + 2^{\frac{n}{3}}(X_2Y_3 + X_3Y_2) + X_3Y_3$$

נעזר בהגדרות שנתנו לנו -

$$E = X_1Y_1 + X_1Y_2 + X_2Y_2 + X_2Y_1$$

$$F = X_2Y_2 + X_2Y_3 + X_3Y_2 + X_3Y_3$$

$$G = X_1Y_1 + X_3Y_1 + X_1Y_3 + X_3Y_3$$

נסמן

$$A = X_1Y_1, B = X_2Y_2, C = X_3Y_3$$

$$XY = 2^{\frac{4n}{3}}(A) + 2^n(E - A - B) + 2^{\frac{2n}{3}}(G - A - C + B) + 2^{\frac{n}{3}}(F - B - C) + C$$

סה"כ הצלחנו לבטא את הביטוי ככפל של 6 מכפלות: A, B, C, E, F, g ולכן קיבלנו את נוסחת הנסיגה הבאה

$$T(n) = 6T(\frac{n}{3}) + O(n)$$

לפי מאסטר $n^{log_3 6} = n^{1.63}$
לצערנו לא הצלחנו להוריד את יעילות הפתרון.. שכן עדיף לחלק ל2 ולקבל $n^{1.58}$.
שאלה 2: יהיו A וB אלגוריתמי מיון מבוססי השוואות. הוכח הפרך
א. יתכן שזמן הריצה של A במקרה הגרוע הינו $nlogn(log(logn))$.
ב. במרבית המקרים (כלומר על יותר ממחצית הקלטים בגודל n, האלגוריתם B ירוץ בזמן לינארי.
פתרון: א. הוכחה. ניתן להשתמש במרג' סורט. נחליט שבכל סיבוב נוסיף לולאה שתרוץ $nlognlog(logn)$ פעמים ולא תעשה כלום. מה אכפת לנו? נקבל מיון ב$nlogn$ ואחכ עוד הזמן בלולאה הזו, טוב נו. עובד.
ב. הפרכה: בכל פעם שניגש לעץ הרקורסיה, נראה כי יש $n!$ פרמוטציות אפשריות לפתרון, כלומר ישנם $n!$ עלים. עץ החלטה הוא בפרט עץ מלא ושלם כי כל העלים באותו גובה וכן תמיד יש שני בנים (ללכת על זה או שלא) ולכן יש $n! - 1$ קודקודים פנימיים. סה"כ בעץ הרקורסיה יש $2n! - 1$ ערכים! ולכן גובה העץ יהיה $log(2n! - 1)$ לפחות, וכן $\Omega(log(n!)) = \Omega(nlogn)$, לכן בסתירה. החסם התחתון יהיה $\Omega(nlogn)$ ולא לינארי כמובן. זה נכון כמובן לרוב הקלטים, ובפרט על יותר ממחצית המקרים.
שאלה 3: הוכח שקיים אלגוריתם שבונה מכל סדרה בעלת n איברים לאו דווקא שונים מהתחום $[1, ...n]$ עץ AVL בזמן $O(n)$
פתרון: נשתמש במיון מניה, גודל המערך הוא n ורוצים למיין בתחום $[1, n]$ כלומר $d = n$ ולכן זמן הריצה למיון יהיה $O(d + n) = 2n = O(n)$
כעת יש לנו מערך ממויין, נבחר את החציון במערך. הוא יהיה קודקוד העץ. כעת נבנה ברקורסיה את העץ בכל פעם על החציונים משמאל ומימין (כלומר תת העץ מימין יהיה הערכים שגדולים מהחציון, שם נבחר חציון כבן ונמשיך הלאה.... ובדומה עבור תת עץ משמאל). נטען כעת טענה -
נטען כי $h(n) = \lceil log(n + 1)\rceil - 1$. טענה זו יש להראות באינדוקציה:
בסיס: $n = 1$, $h(1) = \lceil log(2)\rceil - 1 = 0$ ואכן בעץ עם קודקוד אחד גובהו 0.
צעד: נניח שנכון עבור n ונחלק למקרים.
א. n אי זוגי : אזי יש לנו חציון ואז בכל צד בעץ ישנם $\frac{n-1}{2}$
$h(n) = 1 + h(\frac{n-1}{2}) = 1 + \lceil log((\frac{n-1}{2}) + 1)\rceil - 1 = \lceil log(\frac{n+1}{2})\rceil = \lceil log(n + 1)\rceil - 1$ כנדרש.
ב. אם n זוגי: אזי יש לנו שני עצים בגובה שונה. תת העץ הגדול (בה"כ T_1) מכיל $\frac{n}{2}$ערכים.
מכאן שגובה העץ $h(n) = 1 + h(\frac{n}{2}) = 1 + \lceil log(\frac{n}{2} + 1)\rceil - 1 = \lceil log((\frac{n+2}{2})\rceil = \lceil log(n + 2)\rceil - 1$
נשים לב שלכל מספר זוגי $x > 2$ מתקיים:$\lceil logx\rceil = \lceil log(x - 1)\rceil$
כעת נראה כי אכן העץ ישמור על תכונת AVL: אם n היה אי זוגי אזי היה מכל צד $\frac{n-1}{2}$ איברים, ולכן הפרש הגבהים יהיה 0. אם n היה זוגי אזי בה"כ יש יותר בצד ימין וההפרש יהיה -1.
סה"כ יצרנו עץ AVL. כמובן שעלה לבנות את העץ $O(n)$, שכן מיינו ב$O(n)$ ואז רקורסיבית $T(n) = 2T(\frac{n}{2}) + O(1) = O(n)$.

מבחן אלגוריתמים 1 6102 מועד ב שאלה 5

למי שלא מכיר - פלינדרום הוא מחרוזת סימטרית שניתנת לקריאה משני הכיוונים באופן זהה. תת סדרה של מחרוזת נקראת פלינדרומית אם תת הסדרה היא פלינדרום.
בעיית תת הסדרה הפלינדרומית הארוכה ביותר:
קלט: מחרוזת $S[1, ..n]$
פלט: אורך תת הסדרה הפלינדרומית הארוכה ביותר

פתור לפי השלבים הבאים: נאיבי, רקורסיבי (נוסחת נסיגה מתמטית עם הסבר ולא נכונות פורמלי) + תכנון דינמי שכולל סדר המילוי גודל המבנה. יעילות מקום וזמן. האם ניתן לשפר מקום?

פתרון: נתחיל מהנאיבי ־ נספור כמה אפשרויות לתתי סדרות פלינדרומיות. יש 2^n תתי סדרות ולבדוק האם פלינדרומית יקח $O(n)$ ולכן הפתרון יהיה ב$O(n2^n)$. לא בא בחשבון.

נעבור לרקורסיבי ־ נגדיר את הפונקציה $f(i,j)$ שמחזירה את תת המחרוזת הפלינדרומית הארוכה ביותר שנמצאת בטווח $[i,j]$. כמובן יתקיים $1 \leq i \leq j \leq n$. כעת נרצה לחשוב כיצד נזהה פלינדרום ־ אם $S[i]=S[j]$ אזי אנחנו יודעים שאורך הפלינדרום גדל ב2 שכן מצאנו תת מחרוזת, ונמשיך הלאה לאינדקסים הבאים בטווח. אחרת, נתקדם הלאה ממילא ונבקש מקסימלי. נראה זאת כעת:

$$f(i,j) := \left\{ \begin{array}{cc} 1 & i=j \\ 2+f(i+1,j-1) & S[i]=S[j] \\ max\{f(i+1,j), f(i,j-1)\} & o.w \end{array} \right\}$$

הוכחת נכונות: מקרה ראשון והוא $i=j$. במקרה הזה המחרוזת של התו הבודד היא תמיד פלינדרום ולכן נחזיר 1. אחרת, אם $S[i]=S[j]$ מצאנו שני איברים שיכולים להוות תת סדרה פלינדרומית, כלומר אורך תת הסדרה יהיה 2 לפחות, וכעת נתקדם עם i ימינה ועם j שמאלה. מקרה אחרון יתאים לאם לא מתרחשים התנאים הראשונים, שם נרצה פשוט לבחור בין שני אפשרויות: או להתקדם עם i או להתקדם עם j. על האפשרויות ־ נשים מקסימום שכן המטרה היא למצוא את האורך הגדול ביותר של הפלינדרום.

פתרון תכנון דינמי: נמלא מטריצה בגודל $n \times n$ (לא צריך $n+1$ כי אינדקס ראשון הוא 1 ולא אפס). נמלא את כל האלכסון הראשי שלה ב1 שכן מתקיים $i=j$. נראה כי המטריצה מקיימת תמיד $i \leq j$ ולכן היא תהיה מטריצה משולשית עליונה. מתחת לאלכסון נוכל להתעלם או לשים אפסים. כעת ברצוננו למלא את $\frac{n^2}{2}$האיברים שמעל האלכסון. כיצד נעשה זאת? נראה כי בנוסחה יש תלות פעם בעמודה ובפעם בשורה, לכן נמלא באלכסונים בהתאם לאלכסון הראשי, ואז אלכסון משני וכן הלאה, עד שנגיע לאיבר הימני ביותר מעלה במטריצה. זה יהיה האיבר האחרון שנמלא ־ $A[1,n]$. זה גם יהיה הפתרון שאותו נחזיר, שכן זהו תת המחרוזת הארוכה ביותר מאינדקס 1 עד n שיש בה פלינדרום. נראה כי מילוי כל תא בגישה זו יעלה $O(1)$ וכן יש n^2 תאים, לכן גם זמן וגם מקום יעלה לנו $O(n^2)$. נעיר כי ניתן לצמצם את השימוש במקום אם נחליט שנשמור בכל פעם את האלכסון הנוכחי והאלכסון שקדם לו, ואז אנחנו נגיע ל$O(n)$ מקום.

4202 תל אביב ־ מבחן מועד א שאלה 2

א. נתון מערך A באורך n המכיל מספרים שלמים. תארו אלגוריתם הבודק בזמן לינארי בתוחלת האם קיים זוג איברים שונים כך שהמרחק ביניהם במערך שווה למרחק בין ערכיהם. כלומר האם קיימים $i<j$ כך ש $|A_j - A_i| = j-i$.

פתרון: אמרו תוחלת ־ אמרתי טבלת האש. נראה כי הדבר שקול ל

$$j - A[j] = i - A[i]$$

כעת הבעיה הופכת להרבה יותר קלה. נעבור על איברי הקבוצה ונכניס אותם לטבלת האש באמצעות פונקציית גיבוב מושלמת שתמנע התנגשויות בצורה המרבית. יש סה"כ n ערכים כאלו. כלומר לכל $1 \leq i \leq n$ נכניס את $i-A[i]$ לטבלת האש. כעת, נוכל לסמן את הערכים האלו כx_i. המטרה כעת היא למצוא האם קיימים $x_i = x_j$. נעבור על כל איבר בטבלה, נחפש האם קיים איבר בטבלה בערך שלו (שהרי חיפוש ב$O(1)$) סה"כ מעבר על n איברים יעלה $O(n)$. סה"כ בנינו ב$O(n)$ ואז עברנו על n איברים ־ סה"כ סיבוכיות זמן הריצה $2n+c=O(n)$ כאשר c הוא מס' קבועים קטן וקבוע.

נראה כי יתכן שיש מקרה נוסף ובו

$$j - i = -(A_j - A_i)$$

במצב זה

$$j - i = A[i] - A[j]$$

כלומר

$$A[j]+j=A[i]+i$$

זה מצב לא שונה מקודם. נבנה פשוט טבלת גיבוב נוספת ובה נכניס לכל $1\leq i\leq n$ את $A[i]+i$, נסמנם כy_i. צריך לבדוק האם קיימים $y_i=y_j$. כעת, כמו קודם - נעבור על כל איבר בטבלה ונחפש האם הוא קיים בערך שלו, n חיפושים יעלו $O(n)$

סה"כ שני המקרים יכסו את הבעיה. נעיר כי אם נחפש את הערך עצמו הוא תמיד יופיע - לכן לפני שנחפש נוציא את הערך מהטבלה, נזכור אותו, ואז נחפש (הרי אם נחפש ערך שקיים הוא תמיד יופיע). סה"כ סיבוכיות הפתרון תהיה $O(n)$, לינארי כנדרש. גם במקום.

תרגיל בית אלגו 1 5202 3 - שאלה 3

השאלה: יוסי רוצה לעלות בגרם מדרגות מאורך n כאשר יוסי יכול בכל צעד לעלות מדרגה יחידה או שתי מדרגות. תארו אלגוריתם תכנון דינמי אשר סופר את מס' הדרכים השונות בהן יוסי יכול לטפס במעלה גרם המדרגות, נתחו סיבוכיות זמן ומקום. התחילו מנאיבי אחכ נוסחת נסיגה.

פתרון: נגדיר את המדרגות כמערך $A[1,..n]$. בכל פעם יש ליוסי בחירה של 2 אפשרויות שונות ומתבצעות n בחירות, לכן סה"כ ישנן 2^n אפשרויות שונות, מה שיעלה $O(2^n)$ בנאיבי. ננסה לנסח נוסחת נסיגה. נגדיר את $f(i)$ כמס' האפשרויות שיוסי יכול לצעוד במעלה גרם המדרגות בכדי להגיע ממדרגה 1 אל מדרגה i. כמובן ש$1\leq i\leq n$. כעת נרצה לנסות להבין כיצד לנסח את הנוסחה. נתחיל ממקרי בסיס: אם $i=1$ אזי יש אפשרות אחת בלבד, והיא להשאר במקום. אם אנחנו במקרה הכללי, נראה כי בכל שלב נוכל לבצע אחד משניים: לעשות$f(i-1)$ או לחלופין $f(i-2)$. כלומר נשים לב שבהינתן שנרצה להגיע אל מדרגה i נצטרך לסכום את האפשרויות להגיע אל מדרגה $i-1$ וכן אל $i-2$,כיוון שממדרגה $i-1$ נוכל לבצע קפיצה אחת ולהגיע אל i, וממדרגה $i-2$ נוכל לבצע 2 קפיצות ולהגיע אל i. כעת, נראה כי ישנו תנאי נוסף: אם $i=2$, אזי יש שני דרכים להגיע: או שנבצע קפיצה אחת של 2, או שנבצע שתי קפיצות של אחד. סה"כ מתחילים לקבל נוסחת נסיגה שמזכירה את פיבונאצ'י. נראה כי כעת נוסחת הנסיגה תהיה:

$$f(i):=\left\{\begin{array}{cc}1 & i=1\\ 2 & i=2\\ f(i-1)+f(i-2) & o.w\end{array}\right\}$$

הוכחת נכונות: נרצה להראות שנקבל את כל הפתרונות האפשריים עבור המדרגות $[1,i]$. נוכיח נכונות הנוסחה באינדוקציה:

בסיס: $i=1$ אזי אכן יש אפשרות אחת והיא ללכת צעד אחד. $i=2$ ישנן שתי אפשרויות: ללכת צעד אחד ואז עוד אחד, וללכת שני צעדים.

צעד: נניח שהטענה נכונה לכל $i<n$. כעת יהי גרם המדרגות $[1,i]$. אזי האפשרויות שיש בפני כאשר אני רוצה ללכת אל גרם המדרגות i הינם: 1. ללכת על גרם המדרגות $i-1$ ולהוסיף קפיצה אחת 2. ללכת על גרם המדרגות $i-2$ ולהוסיף קפיצה אחת של 2 או לחלופין 2 קפיצות של 1. ראינו כי ללכת בגרם מדרגות $f(i-1),f(i-2)$ לפי הנחת האינדוקציה יניב את כל הפתרונות האפשריים, ולכן סה"כ אם נחבר אותם שכן הסברנו שיש לאחד את הפתרונות של כל אחד מהם . סה"כ הטענה הוכחה - תמיד נקבל את כל הפתרונות.

כעת נעבור לתכנון דינמי: נשמור מערך בגודל n ובו בכל תא $B[i]=f(i)$. נתחיל במילוי לפי נוסחת הנסיגה - נמלא את התא הראשון והשני. (הערה - לא התייחסתי ל$i=0$ שכן לא יתכן ללכת על אפס מדרגות אבל אם נרצה להתייחס זה יהיה פשוט אפס אפשרויות). ואז נמשיך במילוי משמאל לימין. מילוי כל תא יקח מס' קבוע של פעולות (עד 2) ולכן סה"כ מילוי תא בודד יעלה $O(1)$, מילוי כל התאים יעלה $O(n)$ זמן וכן מקום. הפתרון יהיה ב$B[n]$. נראה כי ניתן לצמצם שימוש בזכרון ל$O(1)$ אם בכל רגע נתון i נשמור את $B[i-1]$ ו$B[i-2]$ ונעדכן אותם תוך כדי. סה"כ סיימנו.

1 אלגו 1 מועד חגים 0202

השאלה: הסלון של אלון הינו מטריצה דו ממדית $n\times n$ כך שמשבצות המטריצה מכילות פירורי אבק. שימי הרובוט נכנס למצב ניקיון בסלון בפינה השמאלית העליונה של המטריצה $[1,1]$ ועליו לסיים את הסיור במשבצת הימנית התחתונה $[n,n]$. שימי יכול לצעוד אך ורק בכיוונים למטה, ימינה, או צעד אלכסוני לכיוון ימינה מטה . שימי אוהב לנקות (מי לא?!) ומטרתו לאסוף כמה שיותר פירורי אבק בסלון של אלון. עזרו לשימי לאסוף הכי הרבה פירורים!

בעיית שימי הרובוט:

קלט: מטריצה דו ממדית $D[1,...n][1,...n]$ של מס' טבעיים כך שהמספר בכל תא מייצג את מס' פירורי האבק.

פלט: מספר פירורים במסלול החוקי שמכיל הכי הרבה פירורי אבק.

פתרו את הבעיה בתכנון דינמי. (אין חובה לנאיבי אך כרגיל מומלץ לקבל אינטואיציה)

פתרון: נרצה לנסות נאיבי ולהשתכנע שזה יותר מדי אפשרויות. בכל רגע נתון (חוץ ממקרי הקצה, מילא) יש לשימי 3 אפשרויות היכן ללכת. לכן נשתכנע שמדובר בסדר גודל של 3^n אפשרויות, אקספוננציאלי, גם נצטרך בכל פעם לעבור על המסלול ולחשב מה שיביא בערך $O(3^n n)$ אפשרויות. נאיבי - לא יעיל - ונתקדם הלאה.

נרצה להגדיר את הבעיה בצורה רקורסיבית. נראה כי בכל רגע נתון אין לשימי יותר מדי אפשרויות. לך ימינה, מטה, או באלכסון ימינה מטה. נגדיר את הפונקציה $f(i,j)$ שתחשב את הסכום המקסימלי ביותר כאשר נסכם למסלול חוקי במעבר מ$[1,1]$ אל $[i,j]$. נשים לב למקרי הקצה: ראשית, אם בטעות ננסה לחרוג מגבולות המערך. כלומר אם נקבל $i>n$ (הלכנו יותר מדי ימינה) או $j>n$ (הלכנו יותר מדי מטה). במקרה כזה נחליט שנגדיר $-\infty$ ונפסיק את הריצה. מקרה נוסף הוא כאשר הגענו ל$i=n$, כלומר אנחנו הגענו לשורה התחתונה, ונניח כי $j\neq n$, אנחנו כעת נוכל ללכת רק ימינה! שכן למטה יביא לחריגה וכנ"ל אלכסון. מקרה נוסף הוא כאשר הגענו לעמודה $j=n$, שם אנחנו נראה שנוכל ללכת רק למטה בשביל להמנע מיציאה מגבולות. אחרת, פשוט החזר את המקסימום משלושת האפשרויות. ננסח את נוסחת הנסיגה כעת פורמלית מתמטית -

$$f(i,j)=\left\{\begin{array}{cc} D[1,1] & i=j=1 \\ -\infty & i>n \\ -\infty & j>n \\ D[i,j]+f(i,j-1) & i=n \wedge j\neq n \\ D[i,j]+f(i-1,j) & j=n \wedge i\neq n \\ max\{f(i,j-1)+D[i,j],f(i-1,j)+D[i,j],f(i-1,j-1)+D[i,j]\} & o.w \end{array}\right\}$$

כעת נעבור לשלב התכנון הדינמי: נגדיר מטריצה B בגודל $n\times n$ (שוב, אין אינדקס אפס לכן לא צריך גבולות של $n+1$). ונתחיל למלא את המטריצה כך - ראשית נמלא את $B[1,1]$ לפי נוסחת הנסיגה. כעת אנחנו נתחיל במילוי לפי שורות משמאל לימין (לפי תנאי הרקורסיה). נראה כי מילוי כל תא מבצע מס' השוואות סופי וקבוע ולכן מילוי כל תא יקח $O(1)$ זמן. יש n^2 תאים ולכן סה"כ זמן וכן מקום יקח $O(n^2)$. נראה כי נוכל להקטין את סיבוכיות המקום אם נחליט בכל פעם לשמור 2 שורות אחרונות, ואז סיבוכיות המקום תהיה $O(n)$.

מבחן תל אביב 2202 מועד א' שאלה 1

השאלה: מגיעים n מפתחות שונים $a_1,...a_n$שהם מספרים שלמים בתחום גדול וידוע מראש,זה אחר זה. כשמגיע a_i יש לענות לפי הגעת המפתח הבא a_{i+1} אם קיים $1\leq j\leq i$ כך ש$(2a_j+3a_i)^3=T$. ניתן להניח כי קיים מספר שלם שהוא חזקה של שלוש. תארו מבני נתונים שתומך בנ"ל בזמן קבוע בתוחלת עבור הכנסה של מפתח. הניחו כי n ידוע מראש ושניתן לאתחל מערך אפסים בזמן קבוע.

פתרון: אמרו תוחלת, אני שמעתי טבלת האש. נבנה טבלת האש. אתחולה יעלה בזכות הנתון זמן קבוע, שכן נאתחל אותה ממערך אפסים. סה"כ האתחול יעלה $O(1)$. כעת נשים לב כי נתון כי T חזקה שלישית של מספר, כלומר קיים b כך ש$T=b^3$ כך שb שלם. כעת הבעיה שקולה להאם קיימים

$$(2a_j+3a_i)^3=b^3$$

שקול לחלוטין

$$2a_j+3a_i=b$$

שקול לחלוטין ל

$$a_j=\frac{b-3a_i}{2}$$

כעת, אנחנו הולכים לפעול כך: נכניס את האיבר הראשון, תמיד נחזיר לא קיים כי הוא הראשון. אחרת, כלומר להכנסות שגדולות מהאיבר הראשון, נרצה לבדוק בטבלת האש האם קיים איבר $\frac{b-3a_i}{2}$. בדיקה שכזו עולה לנו $O(1)$ בטבלת האש. אם אכן קיים נחזיר אמת, אם לא נחזיר שקר. בכל מקרה נכניס את האיבר הבא ב$O(1)$ וסיימנו.

מבחן תל אביב 2202 מועד א' שאלה 3

נתון מערך בגודל n המכיל n איברים שונים. בנוסף נתון כי עבור כל i בין 1 לn האיבר הi בגודלו נמצא באחד המקומות $i-k, i-k+1, ...i+k$ (אם קיימים מקומות כאלו במערך) ($2k$ מקומות). עבור פרמטר k כלשהו בין 1 לn. כלומר, מרחק האיבר ממקומו הממויין לא עולה על k. א. הציעו אלגוריתם שממיין את המערך בזמן $O(nlogk)$ במקרה הגרוע.

פתרון: נשמע קשה אז נעבוד לאט. המרחק של כל איבר ממקומו הממויין הוא k במקסימום. נראה כי לפי ההגבלות, האיבר הקטן ביותר יהיה ב$k+1$ האיברים הראשונים. לשם כך נבנה ערימת מינימום בגודל $k+1$. הבנייה תעלה $O(k)$. כעת נוציא את איבר המינימום ב$O(logk)$ כפי שלמדנו בהרצאה. לפי אותו רעיון ־ האיבר השני בגודלו המינימלי יהיה ב$k+2$ הראשונים, לכן לאחר שנוציא את האיבר המינימלי, נכניס את האיבר ה$k+2$. נשים לב שכך נמשיך בתהליך ־ בכל רגע נתון יהיה בערימה $k+1$ איברים. מתי נפסיק להכניס חדשים? אם הכנסנו כבר את האיבר האחרון. כעת אנחנו כפי שהסברתי בכל פעם: נוציא את המינימלי $O(logk)$, נכניס איבר חדש $O(logk)$. זה יעלה סה"כ $2logk$. במקרה הגרוע נבצע זאת n פעמים (אם $k=0$) או $n-1$ אם $k=1$, בנוסף עלות הבנייה של הערימה עלתה $O(k)$. סה"כ עלה לנו $k+2nlogk = O(nlogk)$. כנדרש.

סעיף ב: הוכח חסם תחתון של $\Omega(nlogn)$ למיון כאשר $k=\sqrt{n}$.

פתרון: נראה כי המיון עלה לנו קודם לכן

$$nlogk = nlog\sqrt{n}$$

קיבלנו כי המיון מתבצע ב

$$O(nlog\sqrt{n}) = O(nlogn^{0.5}) = O(\frac{1}{2}nlogn) = O(nlogn)$$

כעת ניתן להבין שהכוונה הייתה להוכיח חסם תחתון ולא להתבסס על הטענה מההרצאה. נסתכל על עץ ההשוואות של הבעיה. במקרה הגרוע, כל איבר נמצא עד $\sqrt{n}$ מקומות ממקומו המיועד. אם ניקח מערך ממויין ונחלקו לבלוקים של $\sqrt{n}$, נסדר בלוק בכל פרמוטציה אפשרית $\sqrt{n}!$ אפשרויות, לכן נקבל לפי סעיף א' אכן מערך ממויין, סה"כ כל עץ ובחירה שונה תוביל לעלה אחר בעץ החלטה, ונקבל שיש $(\sqrt{n}!)^{\sqrt{n}}$ עלים. נראה כי נצטרך לפחות

$$log((\sqrt{n}!)^{\sqrt{n}}) = \Omega(nlogn)$$

שאלה 4 1102 טומי מועד ב'

השאלה: נתון מערך A של n מספרים כך ש$A[1], ...A[k]$ הם חיוביים ו$A[k+1], ...A[n]$ הם שליליים. עבור $1 \leq k \leq n$. צריך למצוא את k. תאר אלגוריתמים שיעבדו בזמן:

א. $\Theta(logn)$

ב. $\Theta(min(logn, k))$

ג. $\Theta(logk)$

פתרון:

א. נבצע חיפוש בינארי כאשר בכל שלב נגדיר $X = mid$ על האיבר האמצעי. אם האיבר האמצעי הינו חיובי, אזי שהשבירה מתרחשת בצד ימין ולכן נלך ל$[mid+1, ...n]$, אחרת האיבר האמצעי שלילי לכן השבירה מתרחשת כבר בצד שמאל לכן נלך ל$[1, ...mid]$. סה"כ $T(n) = T(\frac{n}{2}) + O(1) = \Theta(logn)$. כנדרש.

ב. כעת נרצה לרוץ על הזמן המינימלי מבין $logn$ לבין k. לשם כך נרוץ במקביל על המערך ־ מצד אחד נרוץ עם האלגוריתם של סעיף א'. מהצד השני נרוץ על איברי המערך מהאיבר ה1 עד לרגע בו נמצא מצב בו האיבר הנוכחי שלילי (שהרי האיברים משמאל בהתחלה אמורים להיות חיוביים). סה"כ הריצה תעצר כאשר נמצא את k, מתי תעצר? בזמן המינימלי משתי האפשרויות. ברור כי אם הריצה תעצר לפי האלגוריתם השני, עברנו על k איברים ולכן הסיבוכיות הינה $\Theta(k)$.

ג. כעת אנחנו נדרשים לעבור על מספר פעולות שגודלן כגודל לוגריתמי של הקלט k. נבצע ריצה בקפיצות של 2 באופן הבא: נתחיל מהאיבר הראשון, נעבור לשני, אח"כ לרביעי, אח"כ לשמיני וכן הלאה... בכל שלב נבדוק האם האיבר הנוכחי הינו חיובי או שלילי. אם מצאנו איבר שלילי באינדקס 2^{i+1}, הרי ש$A[2^{i+1}] < 0$ בעוד $A[2^i] > 0$. כלומר האינדקס k יקיים

$$2^i \leq k \leq 2^{i+1}$$

מכאן

$$2^{i+1} = 2^i * 2 < k * 2 = 2k$$

כלומר

$$2^{i+1} < 2k \Rightarrow i+1 < log(2k)$$

סה"כ, מס' המעברים שעשינו היה חסום ב$O(logk)$, כעת נצטרך לבצע חיפוש באורך של $[2^i, 2^{i+1}]$, החיפוש שם מהסבר דומה יעלה $O(logk)$, סה"כ סיבוכיות הפתרון הינה $O(logk)$.

שאלה 5 טומי 1102 מועד ב'

השאלה: נתון מערך A לא ממויין. תארו אלגוריתם שרץ בזמן $O(n)$ שמוצא את k האיברים בA הקרובים ביותר לחציון של A. רמז: מותר להגדיר מערך עזר בגודל n.

פתרון: נפעיל אלגוריתם סלקט על האיבר ה$\frac{n}{2}$ (בה"כ...) של A. זה יעלה לנו $O(n)$. ניצור מערך עזר D שבו לכל איבר i נחשב את המרחק שלו מהחציון. נפעיל אלגוריתם סלקט על מערך A שמוצא את האיבר הk בגודלו. כעת, לפי האלגוריתם כל האיברים שגדולים מk יהיו מימין וכל אלו שקטנים מk יהיו משמאל לערך k. כעת, כל שנותר הוא להחזיר את האיברים $D[1], ...D[k]$. אלו האיברים הקרובים ביותר לחציון. הערה: נשמור מצביע לכל אחד מn האיברים שיצביע לאיבר המקורי ששייך למרחק שלו. כשנחזיר את האיברים נחזיר בפועל את האיבר שעליו אנחנו מצביעים, שכן ברצוננו להחזיר את האיבר ולא המרחק מהחציון. סיבוכיות: $O(n)$ לסלקט הראשון, יצירת מערך וחישוב המרחקים $O(n)$, אלגוריתם סלקט שוב $O(n)$, אח"כ נעבור על k האיברים הראשונים בD ונחזירם. סה"כ $O(n)$. כנדרש.

שאלה 1 טומי 1102 מועד ב'

השאלה: נתונים שני מערכים ממויינים A בגודל n וB בגודל m. נתון מס' ממשי X. ברצוננו לדעת האם קיימים $a \in A$ ו$b \in B$ כך ש$a + b = X$. כתוב אלגוריתם שבודק זאת בזמן $O(n+m)$.

פתרון:

1. פתרון ראשון - נבנה שתי טבלאות האש. אחת בגודל m והשנייה n ונכניס אליהן את האיברים בהתאמה. כעת נעבור על הטבלה שמתאימה למערך A למשל: עבור כל איבר מn האיברים, נחשב $X - a_i$ ונבדוק בטבלת האש השנייה האם האיבר הזה קיים שם. אם כן - אזי שסיימנו ואכן קיימים שני איברים משני הקבוצות שסכומם X. אחרת, נתקדם לאיבר הבא בטבלה. חיפוש בטבלת האש השנייה יעלה $O(1)$, בנינו שתי טבלאות $n+m$ וכן עברנו על n איברים, סה"כ הפתרון הוא ב$O(m+n)$.
2. פתרון שני - ללא טבלת האש: נאתחל מצביע לאיבר הימני ביותר בA ומצביע לאיבר השמאלי ביותר בB. נחשב את הסכום. אם הסכום יצא גדול מX, אזי שאנחנו צריכים להוריד קצת מהסכום שלהם. לכן נקדם את המצביע בA שמאלה (נקח ערך קטן יותר). אם הסכום יצא קטן מX, נרצה ערכים גדולים יותר ולכן נקדם את המצביע בB ימינה. סה"כ המקרה הגרוע הוא בו אין ערכים כאלו, נעבור על $m+n$ ערכים ולכן $O(m+n)$.

שאלה 3 טומי 1102 מועד א'

השאלה: נתון מערך בגודל n של מספרים שלמים. נניח שידוע כי מספר הערכים השונים במערך הוא $\frac{n}{logn}$ (בערך עליון). תאר אלגוריתם למיון המערך שיעבוד בזמן ממוצע של $O(n)$. כאשר כל הקלטים הינם שווי הסתברות. הראה שהסיבוכיות אכן כנדרש.

פתרון: אמרו ממוצע, הסתברות.. אני חשבתי טבלת האש. אנחנו יודעים כי ישנם $n - \frac{n}{logn}$ ערכים זהים במערך. ניצור טבלת האש עם פונקציית גיבוב. נרצה ליצור אחד בגודל $\frac{n}{logn}$. בנייתה מהמערך תקח $O(n)$. נראה כי כתוצאה

מהנתון, נובע שיש בדיוק איבר אחד שיופיע פעמיים במערך. (ויותר, יופיע $n-\frac{n}{logn}$). כאשר נכניס איברים לטבלת ההאש, בפעם הראשונה שנתקל באיבר שכבר הכנסנו כמובן הוא לא יכנס שוב, אבל נזכור מיהו האיבר. כעת אנחנו יודעים מיהו האיבר שחוזר על עצמו $n-\frac{n}{logn}$ פעמים. נסמנו x. נקח את האיברים שבמערך ללא האיבר x ונמיין אותם. ישנם $\frac{n}{logn}$ איברים שונים כאלו. למיין אותם יעלה במיון כלשהו (למשל ערימה או קוויק סורט) ב

$$\frac{n}{logn}*log(\frac{n}{logn})=\frac{n}{logn}*(logn-log(logn))=n-\frac{nlog(logn)}{logn}<n=O(n)$$

כעת יש לנו מערך של האיברים השונים שממומיין, נסרוק אותו ונבדוק היכן יש להכניס את הערכים שחוזרים על עצמם של x, זה יעלה לכל היותר $O(\frac{n}{logn})$ וקיבלנו מערך ממויין.

סיבוכיות: יצירת הטבלה $O(n)$, מיון האיברים השונים $O(n)$, מעבר על המערך הממויין $O(\frac{n}{logn})$, יצירת מערך חדש שיכלול את האיברים השונים+זה שחוזר על עצמו $O(n)$, סה"כ סיבוכיות הפתרון הוא $O(n)$.

שאלה 5 3102 מועד ב' אלגו 1

השאלה: כאשר חברת טלפונים סלולרית רוצה להתקין אנטנות ישנם שני עקרונות מנחים. הראשון הוא שלא יהיו שתי אנטנות קרובות מדי כי אז הקרינה גבוהה והשני הוא שלא יהיו שתי אנטנות רחוקות מדי כי אז ייתכן ויהיה מקומות של חוסר קליטה. בבעיית מיפוי האנטנות הסלולריות על כביש מאורך N (חד מימדי) הקלט הוא מערך $A=(a_1,...a_n)$ כאשר $0=a_1<a_2<...<a_n=N$, חסם תחתון min ועליון max והמטרה היא לבדוק האם קיימת תת סדרה של A: $A'=a_{i_1},.....a_{i_k}$ של A כך שניתן לשים אנטנות על הכביש המהיר בכל המקומות שמרחקם מנקודת ההתחלה של הכביש נמצא בA' כך ש:

1. $a_{i_k}=a_n=N$, $a_{i_1}=a_1=0$
2. $\forall_{1\le j\le k-1}min\le a_{i_{j+1}}-a_{i_j}\le max$

א. תאר אלגוריתם נאיבי שפותר את הבעיה, הראה שזמן הריצה שלו לא יעיל.
ב. תאר אלגוריתם רקורסיבי (די בנוסחת נסיגה מתמטית + הסבר נכונות מילולי)
ג. תאר אלגוריתם תכנון דינמי שפותר את הבעיה. (השתמש כמובן בסעיף ב') + נתח סיבוכיות זמן ומקום + הסבר האם ניתן לחסוך מקום (לא צריך לשחזר פתרון).

פתרון:

א. ישנם בהינתן n איברים 2^n תתי סדרות. נעבור על כל אחת מתתי הסדרות האלו ונבדוק אם שני התנאים מתקיימים, מעבר על סדרה יקח $O(n)$ ולכן סיבוכיות הפתרון תהיה $O(n2^n)$.

ב. נגדיר פונקציה רקורסיבית $f(i)$ שתבדוק האם קיים פתרון לתת הסדרה שמתחילה באינדקס 1 ומסתיימת באינדקס i (וכוללת אותם) ומקיימת את הדרוש. נראה כי תמיד נרצה לקחת תת סדרה מהאיבר הראשון בשל תנאי 1, ולפי אותו תנאי נרצה גם שהאיבר האחרון יהיה כלול. לכן הפונקציה תחזיר $true$ אם $i=n$, אחרת, נרצה לעבור על כל האינדקסים המקיימים $1\le j\le i-1$ ולבדוק האם מתקיים $min\le a_{i_{j+1}}-a_{i_j}\le max$, במקרה זה נרצה לטעון כי $f(j)=true$. נעיר כאן כי אנחנו נתייחס למקרה טריוייאלי אם קיבלנו $i=0$ נחזיר $true$. נקבל את נוסחת הנסיגה הבאה -

$$f(i)=\left\{\begin{array}{cc} true & i=1 \\ true & \exists_{1\le j\le i-1}min\le A[i]-A[j]\le max\wedge f(j)=true \\ false & o.w\end{array}\right\}$$

נכונות: נראה כי בפנינו בסה"כ שלושה שלבים. במקרה הבסיס נגדיר $i=1$ כנכון, אחרת: נרצה לבדוק אם קיים אינדקס שנמצא בין 1 ל$i-1$ כך שההפרש בינו לבין $A[i-j]$ יהיה בטווח, אם קיים נוכל להחזיר $true$ כמובן. נעיר כעת כי הפונקציה למעשה תוכל להחזיר ערכי $true$ גם באמצע הדרך אך זה לא רלוונטי עבורנו כיוון שהתשובה תמצא ב$f(n)$, האם קיימת תת סדרה שמכילה את 1 וn ועונה על התנאים. כעת אם ננסה לפתור את הבעיה בצורה רקורסיבית זה יגמר רע בסיבוכיות לפחות אקספוננציאלית. נשמור מערך בגודל n, נקרא לו B. בכל תא במערך $B[i]=f(i)$. נמלא את המערך משמאל לימין כמובן בהתאם לנוסחת הנסיגה שמפורטת כאן מעלה. נראה כי בכל מילוי תא עלינו לבדוק האם קיים אינדקס שיקיים את אי השוויון שם, כיוון שצריך לבדוק קיים ייתכן וקיים והוא ממש באינדקס 2 (כלומר, נצטרך לעבור על סדר גודל של $O(n)$ איברים), לכן סה"כ מילוי כל תא יקח $O(n)$ במקרה הגרוע. סה"כ מילוי n תאים כפי שהסברתי יקח סדר גודל של $O(n^2)$. סיבוכיות המקום שלנו תהיה $O(n)$ כיוון שהשתמשנו במערך בגודל n. נעיר כי לא ניתן לבצע צמצום במקום - בכל רגע נתון נרצה לבדוק קיים - נצטרך במידת הצורך לעבור על כל התאים שלפנינו, לכן נהיה חייבים לזכור את כל המקום. הפתרון כמובן יהיה ב$B[n]$.

מבנה נתונים מועד א' 9102

שאלה 1

השאלה: נתון מערך A עם n ערכים ממשיים שונים זה מזה $x_1,...x_n$. ומס' ממשי נוסף y. ברצוננו לבדוק האם קיימים 4 אינדקסים במערך (לא בהכרח שונים) כך ש$\sum_{i=1}^{4} x_i = y$.

א. הצע אלגוריתם לפתרון הבעיה ב$O(n^4)$. הראה שאינו יעיל.

ב. הצע אלגוריתם בזמן $O(n^3 logn)$ עם שטח $O(n)$

ג. הצע אלגוריתם בזמן $O(n^2 logn)$ ושטח $O(n^2)$.

פתרון:

א. מדובר בבעיית $2SUM$ מורחבת. ראינו כי ניתן לעשות $2SUM$ ב$O(n)$ כאשר שומרים בטבלת האש את הערכים ולכל ערך מחשבים $y - x_i$ ומחפשים ($O(1)$) אם קיים בטבלה , אח"כ הכללנו את הרעיון ל$3SUM$ כאשר בכל פעם בטבלת האש נבדוק לכל איבר i האם $x_i + x_1 + x_2 = y$ אמרנו שזה שקול להפעיל $2SUM$ על $x_1 + x_2 = y - x_i$, וראינו שזה עלה $O(n^2)$, כעת נציג פתרון שיפתור ב$O(n^3)$. צריך לבדוק האם קיימים $x_i + x_j + x_k + x_w = y$, שקול ל$x_j + x_k + x_w = y - x_i$. נכניס את כל האיברים לטבלאת האש, עבור כל איבר (סה"כ נעבור על n איברים) נפעיל $3SUM$ על $x_j + x_k + x_w = y - x_i$, ברגע שנמצא נחזיר אמת, אם לא מצאנו ועברנו לאחרון נחזיר שקר. סה"כ זה ירוץ ב$O(n^3)$, שזה כמובן $O(n^4)$.

ללא טבלאת האש אפשר לבדוק אינטיואטיבית עם 4 לולאות פור ־ אחת בתוך השנייה, בכל פעם נרוץ i, j, k, w מ1 ועד לn ונבדוק האם קיימת פרמוטציה כלשהי של האיברים כך שאכן מקבלים את y, סה"כ גם כאן $O(n^4)$. הבהרה ־ בכל פעם ב4 הלולאות (בלולאה הפנימית ביותר) נשאל האם $x_i + x_j + x_k + x_w = y$, וכן בכל ריצה אחד הפרמטרים ישתנה.

ב. כעת נרצה לפתור את הבעיה עם שטח וכן עם $O(n^3 logn)$, מה בעצם שינינו כאן, נראה שבכל זאת נרוץ n^2 פעמים ובכל פעם כזו נרוץ $nlogn$. מתי רצים $nlogn$? או כשממיינים, או כשמבצעים n הוצאות/הכנסות/חיפושים בתוך עץ AVL למשל, או לחלופין הוצאות והכנסות לערימה. נרצה ראשית למיין את המערך. נבצע זאת ב$nlogn$. כעת נבצע ריצה על שלוש לולאות, i, j, k. סה"כ עלה לנו $O(n^3)$. בתחילה נכניס את כל האיברים למערך עזר. זה יהיה שטח האחסון $O(n)$. הוא כבר יהיה ממויין. כעת בכל ריצה אנחנו נחשב את $y - x_i - x_j - x_k$. החישובים כאן התבצעו בלולאה שעולה n^3. כעת, כיוון שהמערך שלנו ממויין, נרצה לבדוק האם קיים במערך איבר ששווה ל$y - x_i - x_j - x_k$. אם כן, סיימנו , הוא יסומן x_w ואז אכן $x_w = y - x_i - x_j - x_k$ כלומר $x_i + x_j + x_k + x_w = y$. נראה שחיפוש במערך ממויין יעלה $O(logn)$, ולכן סה"כ: מיון המערך עלה $O(nlogn)$, ואז ביצענו n^3 ריצות שבכל אחת עשינו $logn$, קיבלנו סה"כ $nlogn + n^3 logn = O(n^3 logn)$.

ג. נרצה להשתמש ברעיון דומה. נרוץ הפעם על שתי לולאות, ולפני זה נמיין את המערך. נשמור מטריצה בגודל $n \times n$ ובה נכניס את כל הוריאציות האפשריות של זוג איברים במערך (סכומם ונניח שנוכל לשמור בכל תא גם איזה אינדקסים בחרנו). "נשטח את המטריצה" להיות כמערך בגודל n^2 ונבצע עליו מיון, שיעלה $O(n^2 log(n^2)) = O(n^2 logn * 2) = O(n^2 logn)$זה יעלה לנו במקום $O(n^2)$ וכן לעבור על כל האפשרויות יעלה גם $O(n^2)$. כעת, נרוץ בשתי לולאות i, j. בכל איטרציה נחשב $y - x_i - x_j$. כעת, נלך למערך המטריצה המשוטח שלנו, נחפש במערך ב$log(n^2) = 2logn$, כיוון שהמערך ממויין, האם קיימים זוג איברים שווים ל$y - x_i - x_j$, אם כן נסמנם x_k וx_w, אזי מתקיים $x_w + x_k = y - x_i - x_j$, כלומר $x_i + x_j + x_k + x_w = y$, כנדרש. וסה"כ נקבל כי עשינו $n^2 logn$ למיון המטריצה, לאחר מכן n^2 איטרציות שבכל אחת מהן עשינו $logn$ פעולות, וסה"כ קיבלנו פתרון ב$O(n^2 logn)$, עם $O(n^2)$ מקום. כנדרש.

שאלה 2

הוכיחו שלא קיים אלגוריתם אשר בהינתן קבוצה עם n מספרים ממשיים בונה בזמן $O(nlog^*n)$ עץ $2-3$ (כלומר $B-TREE$ כאשר $m = 3$) שמכיל איברים אלו.

הוכחה: נב"ש שקיים אלגוריתם A כאשר בהינתן מערך למשל B בגודל n בונה בזמן $O(nlog^*n)$ את העץ $2-3$. כעת נרצה לבצע פעולת $in-order$. נגדיר אותה די זהה לרעיון הכללי בעץ בינארי: $inorder$ בעץ בינארי הולכת לילד השמאלי, אח"כ לערך ואז לערך הימני. נגדיר אותו הדבר בדיוק בעץ $2-3$ רק שכעת המעבר יהיה כך: נלך לילד השמאלי, אח"כ נעלה לערך בין הילד השמאלי לאמצעי, אח"כ נדפיס את הילד האמצעי, אח"כ נעלה לערך השני באבא, ובסוף נלך לערך הימני ביותר. כך נבצע על כל העץ. פעולת $in-order$ כפי שדיברנו והוכחנו בהרצאה מדפיסה את איברי העץ בסדר ממויין. כמו כן, עברנו על n איברים בדיוק ולכן הפעולה לקחה $O(n)$ בדיוק. סה"כ אתחלנו עץ ב$O(nlog^*n)$, ביצענו פעולת $inorder$ שעלתה $O(n)$, וקיבלנו מיון מבוסס השוואות ב$n + nlog^*n$. כלומר, ביצענו מיון ב$O(nlog^*n)$ פעולות, בסתירה. מדוע סתירה? ראינו חסם תחתון למיון מבוסס השוואות $\Omega(nlogn)$ בהרצאה, וכן מתקיים $nlog^*n < nlogn$, כיוון ש$log^*(n) < logn$, ובפרט $O(log^*n) \subseteq O(logn)$, סה"כ קיבלנו סתירה. האלגוריתם לא קיים.

שאלה 3

השאלה: נתון מערך A בגודל n. כל איבר במערך מכיל שני שדות: השדה הראשי מכיל מס' ממשי שהוא הערך של האיבר והשדה המשני מכיל את הערך של האיבר הבא אחרי במערך ממויין (כלומר, מי האיבר הבא שגדול ממנו והכי קרוב אליו). הצע אלגוריתם למיון המערך לפי ערכיו בשדות הראשיים בתוחלת זמן $O(n)$.

פתרון: אמרו תוחלת - נשתמש בטבלת האש כמובן. נכניס את כל האיברים לטבלת האש, בנייתה תעלה $O(n)$. ניצור מערך חדש B בגודל n. נבצע סריקה של המערך ונמצא את האיבר הקטן ביותר (ניתן פשוט לסרוק איבר איבר, אם ממש מתעקשים ניתן לבנות ערימת מינימום ב$O(n)$ ולהוציא את איבר המינימום) . נקח את האיבר הקטן ביותר הזה ונתחיל ממנו, וכמובן אם האינדקס שלו היה $index$ אזי $B[0] = A[index]$. כעת, נסתכל על הערך המשני של האיבר הזה שמצאנו. נלך לטבלאת האש ונחפש אותו בה. חיפוש בטבלת האש יעלה לנו $O(1)$ בתוחלת. כעת קיבלנו את הערך הבא, נכניס אותו למערך B, כעת נלך לשדה המשני שלו, נלך לחפש את האיבר שהוא השדה המשני וכן הלאה. כך נבצע n פעמים ונקבל מערך ממויין. מתי נסיים? כאשר נגיע כמובן ל$B[n]$. נראה כי ביצענו $O(n)$ בהתחלה למצוא את המינימום, ולאחר מכן ביצענו n פעמים מס' קבוע של פעולות סה"כ $O(n)$, לכן הפתרון הוא $n + n = 2n = O(n)$, בתוחלת כמובן.

שאלה 5

השאלה: נכליל את עצי AVL באופן הבא לעצים $AVL3$ טרינריים במקום בינאריים. לכל קודקוד x יש בדיוק 3 תתי עצים (ייתכן כי חלק או כולם ריקים), והפרש הגבהים עבור כל זוג הינו שוב חסום על ± 1 כמו בעץ AVL. בשאלה זו אין צורך להתייחס למפתחות העץ אלא לצורתו הכללית בלבד. יהי $N(h)$ מס' הקודקודים בעץ $AVL3$ מינימלי (עם מס' קודקודים מינימלי). בעל גובה h.

א. רשום את הערכים הראשונים של הסדרה עבור $h = 0, 1, 2, 3$

ב. בנה נוסחה רקורסיבית ל$N(h)$ וחשב בעזרתה את המשך הסדרה מ4 עד ל10.

ג. הוכח שהגבול $lim_{h\to\infty} \frac{N(h)}{N(h-1)} = 2$

פתרון: א. בעץ בגובה 0 יש איבר יחיד, הוא עלה, ולכן $N(0) = 1$. בעץ בגובה 1 כיוון שנתון שלא כל העצים חייבים להיות מלאים, לפחות אחד כן יהיה מלא, ואז נקבל גובה 1, כלומר יש קודקוד ועוד קודקוד וסה"כ $N(1) = 2$, כאשר גובה העץ הוא 2 נראה כי שוב נקח קודקוד, אם ננסה ליצור שרוך שמאלה נקבל שמקדם האיזון של הקודקוד הינו 2, שזה לא טוב, לכן נהיה חייבים להוסיף לעץ הזה גם בנים מימין. כלומר מה שקרה עד כה הוא שיש לנו שורש, עם בן שמאלי ולו בן שמאלי. כעת נוסיף לשורש גם שני בנים ימניים, נקבל כי $N(2) = 5$, וכן העץ האמצעי לפי ההגדרה לא חייב להיות מלא. כעת עבור גובה 3, נראה כי שוב נרצה ליצור שרוך שמאלי, נקבל שם 3 איברים ושוב מקדם האיזון יופר, לכן נוסיף עץ ימני סימטרי אליו, נקבל כי העץ החדש עם ראש,3 בנים ימניים ו3 שמאליים, אך כעת האיזון של עץ הAVL שוב הופר כיוון שלמשל תת העץ השמאלי ביותר (השרוך השמאלי) נמצא בהפרש איזון של 2 עם הבן הימני של השורש. סה"כ נקבל $N(3) = 10$.

ב. ננסה למצוא קשר ממה שקיבלנו בסעיף הקודם. $N(2) = 5 = 2*2+1$, $N(3) = 10 = 2*5$, לא מקדם אותנו. נשים לב כי

$$N(2) = 5 = 2*1+2+1 = 2N(0)+N(1)+1$$

$$N(3) = 2N(1)+N(2)+1$$

זיהינו קשר מסויים, כלומר

$$N(h) = 2N(h-2)+N(h-1)+1$$

נרצה להסביר ולהוכיח מדוע זה נכון. איך בונים עץ עם מס' קודקודים מינימלי שיקיים תדרוש? אנחנו צריכים את הקודקוד (משם מגיע הפלוס אחד), אנחנו צריכים שני תתי עצים שיהיו בגובה $h-2$, בה"כ אלו ימני ושמאלי, ונרצה עץ אמצעי בגובה $h-1$. נשים לב שבמקרה כזה אכן מקדם האיזון בין תת העץ השמאלי לאמצעי הוא יהיה -1, בין האמצעי לימני 1, ובין הימני לשמאלי בדיוק 0. סה"כ אכן הנוסחה נותנת מבנה של עץ AVL. נב"ש שהיא לא נותנת את מס' הקודקודים המינימלי. כלומר, יש קודקוד מיותר בעץ (וייתכן יותר) שלאחריהם נקבל פתרון מינימלי. נסמנו x. בה"כ הוא נמצא בעץ הימני T_R. נוציא אותו מהעץ. קיבלנו סתירה לשלמות הAVL. מדוע? אם הוצאנו אותו מהעץ יכל לקרות שני דברים: 1. הורדנו את גובה העץ - כלומר לקחנו עלה, במקרה כזה נקבל עץ ימני בגובה $h-3$ ועץ אמצעי בגובה $h-1$, הפרש 2 ביניהם וזו סתירה. אפשרות 2. הורדנו איבר כלשהו במסלול - אזי לפי אותו עקרון,

הוא היה עלה במסלול שלו וקיבלנו סתירה. סה"כ אכן הנוסחה מתארת פתרון אופטימלי. חישוב האיברים - כל ילד בכיתה ח' שיודע לפתור נוסחאות נסיגה יכול לעשות. נוותר.

ג. זה לא בחומר שלנו.

9102 מועד ב' טומי

שאלה 1

הציעו מבנה נתונים שתומך בפעולות הבאות: הכנסה, מחיקה, עוקב k- כלומר, אם היינו מחזיקים את הערכים במערך ממויין אזי בהינתן ערך x שנמצא במבנה ומספר k אנחנו רוצים להחזיר את האיבר שנמצא k מקומות מימין לx במערך הממויין. יש לתאר מבנה נתונים שתומך בשלושת הפעולות בזמן של $O(logn)$ לכל פעולה.

פתרון: הבנייה תהיה של עץ AVL וכך נקבל למעשה הכנסה ומחיקה ב$O(logn)$. כעת עלינו לטפל בפעולה "עוקב k". בכל ערך בעץ נחזיק גם לכל קודקוד v את $size(v)$. ראינו כבר בתרגול שאם נוסיף זאת לעץ, ההכנסה והמחיקה לא ישתנו מ$O(logn)$ כיוון שמס' הפעולות שנצטרך לשנות בגלגולים בכדי לשנות את הערך $size$ הינו קבוע. בנוסף, יוחזקו הערכים $size(v_L)$ וכן $size(v_R)$ על מנת לזכור ולדעת את מס' הערכים בצד ימין של העץ ובצד שמאל של העץ בכל קודקוד נתון. כעת אנחנו נחפש את האיבר x. נרצה להתחיל להעלות למעלה ולזוז הצידה בעץ k מקומות. כעת נרצה לזוז k מקומות בצורה חכמה. נבדוק תת עץ ימני של עץ נוכחי. אם גודל תת העץ הימני גדול שווה מk, זה אומר שk נמצא שם. נחפש שם את k. אחרת, אם תת עץ הימני קטן שווה מk, כלומר בהכרח האיבר k לא נמצא בתת העץ הימני. נחסיר מk את $size(v_R)$, נעלה לאבא ונמשיך משם. כאשר הולכים לאבא צריך לזכור מאיזה כיוון הגענו. אם הגענו מהבן השמאלי, אז צריך לבדוק את האבא בעצמו גם כן. אם הגענו מהבן הימני, בהכרח האבא לא מועמד. צריך להמשיך עוד למעלה בעץ. בסה"כ נקבל סיבוכיות של $O(logn)$. מדוע? לכל היותר נעבור על גובה העץ ועוד גובה תת עץ שבהכרח קטן מ$O(logn)$.

שאלה 2

השאלה: נתונים n משתנים $x_1,...x_n$ עם m אילוצים: אילוצי שוויון $x_i = x_j$ ואילוצי אי שוויון $x_i \neq x_j$. נרצה לדעת אם קיים פתרון לבעיה, כלומר תחת m האילוצים האם קיימת השמה למשתנים כך שכל האילוצים יתקיימו. למשל - עבור $x_1 = x_2, x_2 = x_3, x_1 \neq x_3$ לא קיימת השמה כי משני השוויונות נובע $x_1 = x_3$, בסתירה לנתון שאינם שווים.

א. 10 נקודות - הצע אלגוריתם הבודק אם קיימת השמה למשתנים אשר מקיימת את כל האילוצים בזמן ריצה של $O((m+n)log^*n)$ והוכיח נכונותו.

ב. 10 נקודות - הציעו אלגוריתם הבודק אם קיימת השמה למשתנים בזמן ריצה כולל של $O(m+n)$ והוכח נכונותו.

פתרון:

א. רואים log^*n, חושבים $union-find$. למעשה העלות של פעולת $find$ שמחזירה את הקבוצה לה שייך האיבר היא $O(log^*n)$. הדרישה מאיתנו היא לכתוב אלגוריתם שרץ בזמן $O(mlog^*n+nlog^*n)$. ננסה לחשוב על כיוון - אנחנו רוצים לבצע m פעמים פעולת חיפוש בכל ה"עולם", ואח"כ עוד n כאלו. לשם כך, נגדיר את העולם שלנו U כאוסף האיברים $x_1,...,x_n$. כמובן ש$|U| = n$, נרצה ליצור m קבוצות למעשה שיכילו את האילוצים שנדרשים מאיתנו, ולאחר מכן לנסות לבדוק האם האילוצים מתנגשים זה בזה. נתחיל כך - ניצור עולם U שמורכב מכל האיברים. נבצע $UNION$ על כל שני איברים שמקיימים $x_i = x_j$. כעת נעבור על כל אילוצי האי שוויון $x_i \neq x_j$. אם התקיים $Find(x_i) = Find(x_j)$ אזי שמצאנו את שני האיברים שאמורים לקיים אי שוויון שנמצאים באותה קבוצה בה יש שוויון בסתירה. סה"כ נעבור כל על הכל ואם לא נתקלנו במקרה כזה נחזיר שאכן קיימת השמה. מה זמן הריצה? נראה כי במקרה הגרוע, ביצענו $O(n)$ לאתחול, $O(m)$ ל$Union$ וכן אח"כ ביצענו m פעמים log^*n. סה"כ $n + mlog^*n = O((m+n)log^*n)$.

ב. לא בחומר.

שאלה 3

השאלה: נתונים m מערכים ממויינים $A_1,...A_m$. המכילים ערכים שונים זה מזה. נסמן $n = \sum_{i=1}^{m} |A_i|$. הציעו אלגוריתם שמחזיר את המערך הממויין של כל הערכים במערכים המקוריים בזמן ריצה של $O(nlogm)$.

פתרון: ניצור ערימה בגודל m. נכניס אליה את האיברים $A_i[1]$ לכל $1 \leq i \leq m$. בניית ערימה תעלה כגודל הקלט, כלומר $O(m)$. כעת, בכל שלב נוציא את איבר המינימום. זה יעלה $O(logm)$. נכניס לערימה את האיבר הבא (השני בגודלו, אח"כ השלישי וכן הלאה) שמגיע מאותו מערך שהוצאנו ממנו את האיבר קודם לכן. הכנסה לערימה תעלה $O(logm)$. סה"כ נבצע פעולה זו עד שיתרוקנו המערכים, כלומר נעשה זאת על n איברים, לכן מדובר על סיבוכיות של $m + 2logm * n = O(nlogm)$.

הוכחת נכונות: המערכים ממויינים. בשלב הראשון נכנסים לערימה m איברים, כאשר כל אחד מהם הוא האיבר הקטן ביותר במערך שלו. בהכרח לאחר שנשתמש בערימת מינימום נוציא בשלב הראשון את האיבר הקטן ביותר בכל המערכים, שהרי בחרנו מינימום מקבוצת המינימומים. כעת נכניס את האיבר הבא אחריו, שוב קיבלנו ערימה שבה אנחנו מחזיקים את האיבר הכי קטן ביותר מכל קבוצה. באופן דומה, שוב נוציא את המינימום שבמינימומים. כיוון שאנחנו מבצעים n הוצאות של המינימום שבמינימומים נקבל מערך ממויין. כנדרש.

שאלה 4

השאלה: נתון מערך ובו כל איבר יכול להופיע מס' פעמים. צרו ממנו קבוצה, כלומר כל איבר יופיע פעם אחת בלבד, בתוחלת זמן $O(n)$.

פתרון: אמרו תוחלת, כבר גילו לי חצי מהפתרון, אני חושב טבלת האש. נרצה לבנות טבלת האש בגודל $O(n)$. הבנייה תעבוד כך - נתחיל בהכנסת האיבר הראשון. לאחר מכן נתקדם לאיבר השני, לפני שנכניס אותו נחפש האם כבר קיים בטבלה. חיפוש בטבלת האש עולה $O(1)$ בתוחלת. אם קיים - לא נכניס אותו ונתקדם לאיבר הבא במערך - אם לא קיים, נכניס אותו. סה"כ נבצע זאת n פעמים, כאשר בכל שלב נעשה פעולות שעולות $O(1)$. לבסוף קיבלנו טבלת האש שמכילה את האיברים ללא חזרות, כלומר קבוצה, מה שעלה כמובן $O(n)$ בתוחלת.

מבחן במבני נתונים - תל אביב 1202 מועד א'

שאלה 1

נתון מערך A עם n מספרים שלמים. רוצים למצוא את כל הזוגות כך שההפרש בניהם הוא בדיוק 16.

א. ממשו אלגוריתם יעיל בתוחלת שמוצא את כל האינדקסים השונים כך ש$|a_i - a_j| = 16$

ב. נתון בנוסף כי המספרים הם שלמים בתחום $[1, n^3]$. הצע אלגוריתם דטרמיניסטי יעיל ביותר.

פתרון: א. אמרו תוחלת נבנה טבלת האש. הרעיון יהיה ב$O(n)$ שכן חייבים לעבור על הקלט לפחות פעם אחת. נראה כי בהינתן התנאי הדרוש יתכן שההפרש בין שני מספרים יהיה 16 או -16. נבנה טבלת האש מכל האיברים. יעלה $O(n)$. לכל איבר x_i נחשב את $16 - x_i$ ואת $-16 - x_i$ ונחפש האם קיים איבר בטבלה ששווה לאחד מהם. אם כן, מצאנו שני אינדקסים שההפרש בניהם הוא כדקלמן ונחזיר. כמובן שבכל פעם נבדוק לפני האם האינדקס j שמצאנו שונה מi. סה"כ נעבור על n איברים, חיפוש עולה $O(1)$ בתוחלת ולכן זה יעלה $O(n)$, בתוספת הבנייה $n + n = O(2n) = O(n)$ בתוחלת.

ב. נרצה לנסות למיין אותם מה שיעזור לנו. ננסה להבין כמה ספרות יש במספר: $d = \lceil log_b(n^3) \rceil = \lceil 3log_b n \rceil$, ב$radix$ מיון הסיבוכיות היא $O(d(n+b))$. אם נבחר את הבסיס b להיות n אזי נקבל כי $d = 3log_n n = 3$, וכעת נקבל סיבוכיות $O(3(n+n)) = O(6n) = O(n)$, כלומר נצליח למיין ב$O(n)$ את המערך שלנו. כעת כיצד זה עוזר לנו? האיברים מסודרים בסדר עולה, כעת נעבור ב2 אינדקסים על המערך הממויין. נאתחל אינדקס $i = 1$ וכן $j = n$. אם ההפרש יצא גדול מ16, נעשה $j - -$. אם ההפרש קטן מ16, צריך להוסיף איברים, נעשה $i + +$. סה"כ גם כאן עשינו ב$O(n)$.

שאלה 2

בשאלה זו נרצה לבנות מבנה נתונים Q אשר יתחזק אוסף של זוגות שונים של מספרים (a, b) (ייתכנו שוויונות בערכי a או b בין זוגות שונים של איברים אך לא יתכן בו זמנית שיהיה שני זוגות זהים), נרצה שמבנה הנתונים יתמוך בפעולות הבאות:

א. $inser(a, b)$ = מכניסה למבנה הנתונים את הזוג (a, b)

ב. $Delete(a, b)$- מקבלת מצביע לאיבר (a, b) ומוחקת אותו ממבנה הנתונים

ג. $QueryMin(x, y)$- מחזירה זוג (a, b) שמקיים $b = y$ וכן $a \geq x$, וכן a הוא המינימלי שמקיים זאת. אם לא קיים זוג כזה החזר $null$.

1. הצע מבנה נתונים שמממש את כל הפעולות ב$O(logn)$

2. במקום הפעולה $QueryMin$ נרצה לממש את הפעולה הבאה - $QueryMax(y)$, מחזירה זוג (a, b) כך שa מקסימלי מבין המפתחות המקיימים $b = y$ ואם לא קיים יש להחזיר $null$. הצע מבנה נתונים שמממש את א+ב ב$O(logn)$ בתוחלת ואת $QueryMax$ ב$O(1)$ בתוחלת.

פתרון:

א. נרצה להשתמש בעץ AVL וכך נקבל די "בחינם" הכנסה ומחיקה ב$O(logn)$. נעיר שסדר הגודל בעץ יהיה לפי האיבר b, ולא לפי a. מה שנותר לטפל בו הוא $QueryMin$ כמובן. נרצה ראשית לבדוק האם קיים זוג בו $b = y$. כיוון שזה עץ AVL לפי b, נוכל לחפש ב$O(logn)$. אם לא מצאנו כזה כבר נחזיר $null$. כעת, נניח שאכן מצאנו כזה, כל שנותר לנו הוא לבדוק האם $a \geq x$. נשים לב שייתכנו כמה זוגות כאלו. אבל היכן יהיו הזוגות הללו? כאן אנחנו נחליט על הוספת עץ T_b שיחזיק בכל רגע נתון את כל האיברים a שמקיימים שהזוג שלהם $b' = b$. נראה שזה לא ישפיע יותר מדי על המחיקה וההכנסה לעץ, שכן אם נמחק או נכניס נוסיף גם לעץ הזה וזה יעלה $O(logn)$ שהרי $|T_b| \leq n$ גם כן.

כעת, לאחר שמצאנו את הערך b הרצוי נסתכל על העץ שלו T_b, שם נחפש את האיבר המינימלי ביותר a בעץ הזה שגם מקיים $x \geq a$. נסתכל בעץ T_b על האיבר הראשון, אם הוא גדול כבר מאיקס, נרצה ללכת שמאלה בעץ לבדוק אם יש קטנים ממנו. אם הוא קטן מאיקס, מי שיכול להיות גדול מאיקס בהכרח נמצא מימין. כך למעשה נבצע חיפוש בינארי על העץ למציאת האיבר a הנ"ל המינימלי. לבסוף נמצא אותו בעד $O(logn)$. סה"כ פעולות ההכנסה והמחיקה יעלו עד $2logn = O(logn)$ וכן פעולת ה$Query$ תעלה $logn$ לחיפוש b כנ"ל ועוד עד $logn$ למציאת a המינימלי אם קיים (אם לא נחזיר שלא קיים), וסה"כ גם $O(logn)$.

ב. אמרו תוחלת נשתמש בטבלת האש הפעם. אנחנו נשמור את כל המפתחות בתוך טבלת האש שתשמש גם כמילון. הטבלה תשמש לפי B, כלומר המפתח יהיה b ולו רשימה של כל המפתחות a שקשורים אליו. לכל ערך b כזה יהיה ערימת מקסימום עם ערכי a. איך נממש הכנסה של הזוג? נמצא את המפתח b המתאים ב$O(1)$ ואז נוכל לגשת לערימת המקסימום שלו, והכנסה לערימת מקסימום היא ב$O(logn)$, ושוב בתוחלת כי אנחנו בהאש. איך נמחק זוג? שוב נמצא מפתח b ב$O(1)$, ואז נמחק מערימת מקסימום ב$O(logn)$. כעת הפעולה הקשה מכולם תהיה קלה - נחפש האם קיים $b = y$ ב$O(1)$, אם כן, נשלוף את הערך מערימת המקסימום שלו, שכן רצו את הa המקסימלי שמקיים $b = y$. קליל.

שאלה 3

הצע אלגוריתם יעיל בתוחלת הממיין n מספרים שלמים כאשר ידוע שיש רק $\frac{n}{logn}$ מספרים שונים מבין n המספרים (בפרט יש חזרות)

פתרון: ידוע כי יש $\frac{n}{logn}$ איברים שונים. נבנה טבלת האש (אמרו תוחלת) ואליה נכניס את האיברים, נשים לב שלפני כל הכנסה נבצע חיפוש האם כבר קיים בטבלת האש ערך כזה, חיפוש ב$O(1)$, ולכן נקבל טבלת האש של כל המספרים ששונים זה מזה. בכל שלב כאשר נרצה להכניס ערך שכבר קיים, נוסיף $+1$ לקאונטר שיחזיק את מס' הערכים שקיימים מאותו ערך, שיהיה איזשהו שדה לכל איבר בטבלה. כעת נבנה מהטבלה מערך (פשוט נמיין אותה) ב$klogk$ כאשר $k = \frac{n}{logn}$. נקבל

$$O(\frac{n}{logn}log(\frac{n}{logn})) = O(\frac{n}{logn}(logn - log(logn)) = O(n - \frac{nlog(logn)}{logn}) = O(n)$$

כלומר הצלחנו למיין אותה ב$O(n)$, כעת כיוון שאנחנו זוכרים לכל איבר כמה מופעים צריכים להיות ממנו, הדבר יהיה קל. נבנה מערך B כך שיעבור על המערך A ויכניס לפי הסדר x_i ערכים מאותו הערך, קיבלנו מערך ממויין ב$O(n)$.

שאלה 4

א. תכננו מבנה נתונים Q שמתחזק קבוצת מספרים שמתחילה כקבוצה ריקה ותומך בפעולות הבאות:

$insert(Q, k)$ הוספת איבר x עם מפתח k לq

$Delete(Q, x)$ מחיקת האיבר x מQ (כאשר יש מצביע לx)

$ApproxMedian(Q)$= מחזירה מס' כלשהו t (לא בהכרח במבנה) הגדול מלפחות $\frac{n}{4}$ מאיבר Q וקטן לפחות מ$\frac{n}{4}$ מאיבר Q. כאשר n הוא מס' האיברים הנוכחי במבנה.

כל הפעולות צריכות להיות ב$O(1)$ לשיעורין.

פתרון: נלך לכיוון של רשימה מקושרת (אין לוג אז אין עץ, ואין תוחלת אז לא האש.. אין לוגסטאר אז לא יוניון פינד. מה נשאר? פשוט - מערך, רשימה... אולי מחסנית). ננסה רשימה. ככה ניתן להוסיף ולמחוק איבר בקלות עם מצביעים, ב$O(1)$, תמיד, לא רק לשיעורין. כעת נותר לטפל בפעולה האחרונה שנראית די מעצבנת. נרצה בכל שלב נתון להיות באחוזון ה75 עד ה25 של האיברים כרגע.

כיוון שאין צורך בערך t ספציפי, מספיק שנחשב את החציון במערך (הרי הוא אכן עונה על הדרוש) כל $\frac{n}{4}$ פעולות. מדוע? כאשר נוסיף $\frac{n}{4}$ איברים או נוריד, במקרה הגרוע אם הכל היה הוספה החציון יהפוך ממש להיות אחוזון 75, ואם נוריד במקרה הגרוע כלומר הכל היה מחיקה נגיע לאחוזון 25. זה עונה על הדרוש בסה"כ. למצוא את החציון עולה $O(n)$, אבל כיוון שנבצע זאת רק כל $\frac{n}{4}$ פעולות, נרצה לראות שלשיעורין זה $O(1)$. כיצד? מתבצע $O(n)$ כל $\frac{n}{4}$ פעולות, כלומר $\frac{n}{\frac{n}{4}} = 4$, עלות ממוצעת פר פעולה. $O(4) = O(1)$.

אלגו' 1 מועד ב 8102 שאלה 5

נתון מערך A של מס' ממשיים $A = [a_1, ...a_n]$. d־חלוקה היא חלוקה של A לd מקטעים זרים כלומר בחירה של $d - 1$ אינדקסים $1 \leq i_1 < i_2 < ... < i_{d-1}$ שמגדירים את d המקטעים $[a_1, ...a_{i_1]}, ..., [a_{i_{d-1}+1}, ...a_n]$. העלות של מקטע ב$d$ חלוקה היא סכום המספרים באותו המקטע. העלות של d חלוקה היא העלות המקסימלית של מקטע מבין המקטעים שהיא מגדירה. הבעיה -

קלט: מערך A של n מס' ממשיים ומס' שלם חיובי d.
פלט: העלות המינימלית של d חלוקה של המערך A (מבין כלל הd חלוקות האפשריות)
א. כתוב נוסחה רקורסיבית לפתרון הבעיה.
ב. הצג אלגוריתם תכנון דינמי לפתרון הבעיה. נתח סיבוכיות זמן ריצה ומקום.
פתרון:
זה יכול לבלבל לכן נראה שהמטרה היא למצוא את העלות המינימלית, של המקטע המקסימלי בתוך חלוקה. כלומר, המטרה היא למצוא את הmin של החלוקות האפשריות של ה(המקטע שאורכו מקסימלי) - כלומר החלוקה בה הסכום המקסימלי, תהיה החלוקה המינימלית ביותר. אחרי שהתגברנו על העברית ננסה להתקדם -
ננסה לבנות פונקציה, ננסה לחשוב מה אנחנו צריכים בשביל הפונקציה. באופן הנאיבי, היינו עוברים על כל החלוקות (2^n) ומחשבים את הערך של החלוקה המקסימלית $O(n)$ ומתוך זה, היינו לוקחים את המינימלי. סה"כ זה היה עולה $O(n2^n)$. יקר. כעת ננסה להגדיר משהו ונשנה אותו במידת הצורך, נגדיר $f(i,j)$ כעלות המינימלית של החלוקה j (כלומר מס' החלוקות) במקטע $[1,..i]$. כעת, נרצה לעבור על כל האפשרויות השונות "לחתוך" את המקטע לחלקים. כלומר, נחשוב רקורסיבית היכן אנחנו נשים את d בחתך האחרון. אם נשים את החתך באינדקס מסויים, נקרא לו k, כתוצאה מכך המקטע האחרון יהיה $[k+1,...n]$ והשאר $[1,...k]$ צריכים להתחלק ל$d-1$ חלקים. לכן ננסה להגדיר את הנוסחה הבאה:

$$f(i,j) = min\{_{1\leq k<i} max\{f(k,j-1), sum(k+1,i)\}$$

נסביר: בהינתן מקטע איברים $[1,...i]$ וj מקטעים שנרצה לחלק, נרצה לעבור על המינימום מבין כל החלוקות. מהם כל החלוקות? בכל פעם נרצה לבחור את המקסימום מבין להסתכל על החלוקה $[1,...k]$ כאשר $j'=j-1$, כי ירד לנו מקטע, שהיא תניב לי ערך מקסימלי לתת בעיה, או לחלופין לקחת את המקטע האחרון. סה"כ נקבל מתוך זה עבור כל חלוקה את המקסימום האפשרי, כעת, על כל המקסימומים האלו נפעיל min ונקבל את הפתרון. נשים לב לכמה מקרי בסיס - אם $i=1$ אזי מדובר על מקטע של $[1,..1]$ כלומר איבר בודד, במקרה כזה אנחנו נוכל כמובן להחזיר רק את a_1. כעת, פתרון הנוסחה ברקורסיה יעלה אקספוננציאלי. ניצור מטריצה, נרצה ש $B[i,j]=f(i,j)$ לכן מימדי המטריצה יהיו $n\times d$. נשים לב שמילוי כל תא, יעלה $O(n)$ בשל הדרישה לעבור על המקסימום מבין המקטעים. נמלא בעמודות, כל תא יעלה למלא $O(n)$. הפתרון יהיה בתא $B(n,d)$ שיחזיר את הערך המינימלי מבין המקסימומים. סה"כ זמן המילוי יהיה $O(n)$ לתא כפול nd ולכן $O(n^2d)$, וכן סיבוכיות המקום תהיה $O(nd)$. האם ניתן לשפר את סיבוכיות המקום? ננסה להבין מהנוסחה - מה קורה בעצם? אנחנו בכל שלב עושים מקסימום על כל הקודמים, ולכן זקוקים אותם, ולצערנו לא נוכל לחסוך במקום.

אלגו' 1 תרגיל 5 שאלה 1 2025:

מכונית מתוצרת אלגו-קאר נוסעת בהנעה חשמלית. כאשר הסוללה טעונה במלואה, המכונית יכולה לנסוע בדיוק m קילומטרים עד שהסוללה תתרוקן. אנו מתכננים לצאת לנסיעה על כביש ארוך המתחיל בנקודה A ונגמר בנקודה B שפרוסות לאורכו n תחנות טעינה. התחנות ממוקמות במיקומים $p_1<p_2<.....<p_n$ כאשר p_i הוא המרחק של תחתנת הטעינה ה-i מהנקודה A (וכן מתקיים $A<p_1$ וכן $B>p_n$) מובטח לנו כי לכל $1\leq i\leq n-1$ מתקיים $p_{i+1}-p_i\leq m$. הצע אלגוריתם חמדני הממזער את מספר העצירות של הרכב לצרכי טעינה, והוכח את נכונותו.
פתרון: האינטואיציה תהיה פשוטה - בכל שלב נבחר את התחנה הרחוקה ביותר שניתן להגיע אליה בהתחשב במיכל הדלק הנוכחי. נאתחל רשימה ריקה $places$ ואת המיקום הנוכחי שלנו $spot=A$. בכל פעם נבדוק מיהי התחנה האחרונה והרחוקה ביותר p_i שניתן להגיע אליה מהמיקום הנוכחי. כלומר תחנה i כך ש $p_{i+1}-spot\geq m$ וכן $p_i-spot\leq m$, נכניס את p_i לרשימה של $places$ בה אנחנו נבקר ונעדכן את המיקום $spot$ להיות p_i. בסוף, כאשר $p_i=B$ נחזיר את $places$.
מדוע זה מבטיח פתרון?
טענה - קיים פתרון אופטימלי לבעיית תחנות הטעינה , בו בוחרים את התחנה האחרונה ביותר שניתן להגיע אליה מA.
הוכחה - נסמן p_i את התחנה האחרונה ביותר שניתן להגיע אליה מA, נסתכל על פתרון אופטימלי OPT כלשהו. אם $p_i\in OPT$ הרי סיימנו, שכן הוא בפתרון האופטימלי. אחרת, נסתכל על התחנות הראשונות שיהיו קטנות מp_i בOPT. נסמנן $p_{j_1},.....p_{j_r}$. בהכרח, לפי הגדרת p, לכל $1\leq k\leq r$ מתקיים $p_{j_k}<p_i$. נגדיר OPT'. ע"י הוצאת כל p_{j_k}והוספת p_i. כלומר

$$OPT' = (OPT/\{p_{j_1},.....p_{j_r}) \vee (p_i)$$

כעת נוכיח כי OPT' הינו פתרון חוקי ואופטימלי.

חוקי - לכל זוג נק' צמודות בפתרון שנמצאות לאחר p_i לא שינינו כלום וכן לפי ההגדרה $p_i - A \leq m$, והמרחק בניהם תקין לכן חוקי. כמו כן, נבין בתחנה הראשונה ב-OPT' שמגיעה לאחר p_i. נקרא לה p_l , מתקיים $p_l - p_i \leq p_l - p_{j_r} \leq m$, לכן פתרון חוקי כמובן.
אופטימלי - קיימת לפחות p_j אחת, ומתקיים

$$|OPT'| = |(OPT/\{p_{j_1}, ...p_{j_r}) \vee (p_i)| \leq |OPT| - 1 + 1 = |OPT|$$

כלומר אכן גודל הפתרון קטן שווה מהאופטימלי, ולכן בהכרח אופטימלי בעצמו וחוקי.
טענה נוספת - תכונת תת המבנה האופטימלי: פתרון שמורכב מבחירה של התחנה האחרונה ביותר שניתן להגיע אליה בתוספת פתרון אופטימלי לבעיות הטעינה עם כל התחנות שנמצאות אחרי התחנה הנ"ל הוא פתרון אופטימלי.
נוכיח. כעת, נניח בשלילה שהפתרון של האלגוריתם A איננו אופטימלי. כלומר קיים B פתרון כך ש-$|B| < |A|$. עקב הבחירה החמדנית בהכרח $p_i \in B$. היא הראשונה בדרך. בפרט $A/\{p_i\}$ וכן $B/\{p_i\}$ הם פתרונות לתחנות הטעינה עבור הערכים לאחר p_i.בפרט, $A/\{p_i\}$ פתרון אופטימלי. כלומר

$$|A/\{p_i\}| \leq |B/\{p_i\}| \Longrightarrow |A| - 1 \leq |B| - 1 \Longrightarrow |A| \leq |B|$$

בסתירה ל-$|B| < |A|$. סה"כ אכן נקבל פתרון אופטימלי.
זמן הריצה של הבעיה יהיה לינארי שכן מדובר במעבר על הרשימה, לכן $O(n)$.

2025 תרגיל 3 אלגו - שאלה 1

השאלה: נתון לוח בגודל $n \times n$ שבכל משבצת בלוח שמור מספר שלם כלשהו. נגדיר את בעיית משחק הלוח כך: השחקן מתחיל את צעדיו במשבצת $(1,1)$ ומסיים במשבצת (n,n). בכל צעד מותר לו ללכת ימינה או למטה או באלכסון המתאים (ימינה מטה). כלומר, בהינתן שהוא במשבצת (i,j) מותר לו ללכת אל $(i+1,j), (i,j+1), (i+1,j+1)$. כמובן אם אפשרי - אם בקצה הימני אזי בהתאם. בכל צעד שהוא דורך במשבצת הוא מקבל מס' נקודות ששמור במשבצת. מטרת השחקן היא לאסוף כמה שיותר נקודות. תאר אלגוריתם תכנון דינמי לפי השלבים שפותר את הבעיה.
פתרון: נתחיל מהאלגוריתם הנאיבי. בכל שלב יש לו בחירה של 3 אפשרויות לאן להתקדם, אפשר לעבור על כל האפשרויות במסלול ולחשב את המקסימלי, מדברים על $\Omega(n3^n)$, אקספוננציאלי ולא בא בחשבון.
נעבור לאלגוריתם הרקורסיבי. נגדיר פונקציה $f(i,j)$ שמחזירה את הסכום המקסימלי של ערכים שצבר השחקן במעברו מהנקודה $(1,1)$ אל הנקודה (i,j). כמובן, שבכל שלב נתון נחזיר מקסימום מבין האפשרויות. אך נדרש למקרי קצה:
א. אם אנחנו נמצאים בשורה $i = n$, אין לנו ברירה אלא ללכת כל הזמן ימינה.
ב. אם אנחנו בעמודה $j = n$, אין לנו ברירה אלא ללכת בכל פעם למטה.
ג. אם בטעות זלגנו מגבולות המערך, נגדיר $-\infty$ והפתרון ייפסל.
נגיע אל נוסחת הנסיגה הבאה:

$$f(i,j) = \left\{ \begin{array}{cc} A[i][j] & i = j = n \\ A[i][j] + f(i-1,j) & i = n \\ A[i][j] + f(i,j-1) & j = n \\ -\infty & i > n \vee j > n \\ max(A[i][j] + f(i-1,j), A[i][j] + f(i-1,j-1), A[i][j] + f(i,j-1) & o.w \end{array} \right\}$$

מדוע שמנו מינוס? נשים לב שאנחנו ממלאים את המערך מהערך $A[1,1]$ עד ל-n. ולכן אנחנו בכל שלב בוחרים מאיזה כיוון להתקדם. ניתן להוסיף כמובן תנאי עצירה של $f(1,1) = A[1,1]$.
כעת לפתרון הדינמי: נבנה מטריצה בגודל $n \times n$. נתחיל במילוי המטריצה מהנקודה $A[1,1]$. כעת, נמלא את המטריצה לפי שורות מה שיבטיח לנו מילוי כל תא ב-$O(1)$, סה"כ יש לנו n^2 תאים לכן סיבוכיות הזמן תהיה $O(n^2)$ וכך גם סיבוכיות המקום $O(n^2)$. נוכל להוריד את סיבוכיות המקום אם בכל שלב נשמור 2 עמודות קודמות, ונגיע לסיבוכיות של $O(n)$ במקום. הפתרון לכל הבעיה יהיה בתא (n,n) של המטריצה.

תרגיל 4 אלגוריתמים 1 שאלה 1

בהינתן מחרוזת $S = s_1, ..s_k$ נגדיר שלוש פעולות עריכה חוקיות:
א. הוספת אות σ במיקום i. מייצרת את המחרוזת $S' = s_1, ...s_{i-1}\sigma s_i....s_k$

ב. מחיקת אות ממיקום i מייצרת את המחרוזת $S' = s_1, ...s_{i-1}s_{i+1}....s_k$

ג. החלפת אות באות σ במיקום i תיצור את המחרוזת $S' = s_1, ...s_{i-1}\sigma s_i....s_k$.

בהינתן שני מחרוזות A וB נגדיר את מרחק העריכה בתור אורך הסדרה הקצרה ביותר של פעולות עריכה כך שאם מפעילים את סדרת הפעולות על המחרוזת A מגיעים לבסוף למחרוזת B. נרצה פתרון מינימלי למס' העריכות.

קלט: שתי מחרוזות A וB לא בהכרח באותו האורך.

פלט: האורך של סדרה קצרה ביותר של פעולות עריכה שכאשר מפעילים אותן החל מהמחרוזת A מקבלים לבסוף את המחרוזת B.

פתרו את הבעיה בתכנון דינמי ע"פ השלבים שלמדתם.

פתרון:

באשר לפתרון הנאיבי, נוכל לבדוק את כל האפשרויות להחלפה. בכל שלב הרי אנחנו עומדים בבחירה בין 3 אפשרויות לשינוי המחרוזת. בהינתן שאורכי הרשימות הם ℓ_1, ℓ_2 בהתאמה אנחנו נבצע $3^{\ell_1+\ell_2}$ אפשרויות במקרה הגרוע (אם נמחק הכל מA ונוסיף לB). לכן עומדים על פתרון אקספוננציאלי של $\Omega(3^{\ell_1+\ell_2})$ וזה עוד לפני שבדקנו שזה מינימלי.

נעבור לפתרון הרקורסיבי. נגדיר $f(i,j)$ כפונקציה שמחזירה את מס' פעולות העריכה המינימלי עבור לעבור מ$A[1,...i]$ ל$B[1,...j]$. נשים לב למס' מצבים:

בכל רגע נתון אנחנו רוצים לעבוד עם אינדקס של i מA ולבצע עליו 3 מניפולציות.

אם נקבל $i = 0$ למעשה, אזי סיימנו לעבור על כל הרשימה A. נרצה שהיא תהיה זהה לB ולכן נוסיף תווים כלומר $f(i, j-1) + 1$.

אם $j = 0$, סיימנו לעבור על B ויש תווים בשניהם, לכן נמחק תווים מA.

אם $i = 0$ וגם $j = 0$ סיימנו לעבור על שניהם ונחזיר 0.

הוספת אות יהיה למעשה $f(i, j-1) + 1$ שכן הוספנו אות , ולכן כעת אנחנו רוצים להסתכל על המקטע של עד $j - 1$ כי כבר טיפלנו בבעיה.

מחיקת אות, יהיה למעשה $f(i-1, j) + 1$ שכן אנחנו כעת עוברים לבעיה של $i - 1$.

החלפת אות תהיה למעשה ליצור מצב בו $A[i] = B[j]$, לכן תת הבעיה שלנו הופכת להיות לטפל במחרוזות $1,..i$ וכן $1, ...j$

אם $A[i] = B[j]$ אזי התווים זהים ואנחנו לא נדרשים לבעיית עריכה ולכן נצטמצם כמו בהחלפת אות.

נוכל להסתכל על כך בנוסחת הנסיגה הבאה:

$$f(i,j) = \begin{Bmatrix} 0 & i = 0 \wedge j = 0 \\ f(i, j-1) + 1 & i = 0 \\ f(i-1, j) + 1 & j = 0 \\ f(i-1, j-1) & A[i] = B[j] \\ min\{f(i, j-1) + 1, f(i-1, j) + 1, f(i-1, j-1) + 1\} & o.w \end{Bmatrix}$$

לבסוף קיבלנו נוסחה. כעת, ניגש לפתרון התכנון הדינמי. נניח כי גודל A הינו n וגודל B הינו k. נבנה מטריצה בגודל $n \times k$. נרצה להתחיל למלא אותה מהתא $[0, 0]$ וכן למלא את השורה הראשונה $i = 0$ ואת העמודה הראשונה $j = 0$ לפי נוסחת הנסיגה. לאחר מכן, נשים לב שבהינתן תא כלשהו אנחנו זקוקים לתא שנמצא באלכסון שלו מלמעלה, או לתא מימינו או לתא שמעליו. לכן אנחנו נמלא לפי שורות, מה שיבטיח לנו שכבר מילאנו בשורה קודם את מי שמעליו או באלכסונו מהצד שמאל, וכן כיוון שנמלא את השורות משמאל כבר מילאנו את מי שמשמאלו. סה"כ מילוי כל תא יעלה $O(1)$ בשיטה זו. אנחנו עוברים על nk תאים ולכן סיבוכיות הזמן תהיה $O(nk)$ והמקום $O(nk)$. הפתרון יופיע בתא (n, k). נוכל לשפר את סיבוכיות המקום אם נשמור בכל שלב 2 שורות ־ הנוכחית והקודמת, מה שיוריד את סיבוכיות המקום להיות $O(k)$, שכן גודל של שורה הוא k.

2025 תרגיל 5 שאלה 4

השאלה: מתכנת מקבל מחברה רשימה של עבודות, בה לכל עבודה יש מועד אחרון לביצוע ורווח משויך אם העבודה מתבצעת לפני המועד האחרון. למתכנת לוקח לבצע כל עבודה יום אחד בדיוק, והוא יכול לעבוד במקביל רק על עבודה אחת. המתכנת מעוניין למקסם את הרווח הכולל ולבנות לעצמו תוכנית עבודה שתאפשר לו להרוויח כמה שיותר. הצע אלגוריתם חמדני למתכנת החמדני הממקסם את הרווח הצפוי למתכנת ע"י בחירת העבודות הרווחיות ביותר שניתן לבצע לפני המועד האחרון.

למשל ־ עבור רשימת העבודות: עבודה 1 : רווח 40 מועד אחרון 1, עבודה 2: רווח 20 מועד אחרון 1: עבודה 3 רווח 15 מועד אחרון 1 עבודה 4 רווח 40 מועד אחרון 3, ירצה המתכנת את עבודות 1 (לאחר שיקח אותה לא יוכל לקחת אחרות כי מסתיים באותו יום) ואת עבודה 4.

פתרון: ננסה לחשוב על זה כמו בעיה מהחיים ־ וננסה את הגישה הבאה: אנחנו רוצים להרוויח כמה שיותר, למעשה אם כל עבודה לוקחת יום אחד, אנחנו נסתכל על רשימת העבודות ונמיין אותם לפי המועד בו הם מסתיימות, בכל פעם נבחר את העבודה המקסימלית ברווח. אם נעבור על הימים מהראשון לאחרון זה לא יניב פתרון אופטימלי. לעומת זאת, אם נעבור על הימים מהסוף לראשון, כאשר בכל פעם נבחר את הרווח המקסימלי מבין העבודות הדחופות, נגיע

לפתרון אופטימלי. נשים לב כי ייתכן שבדרך יהיו ימים ללא עבודות, למשל כאן בדוגמה, נתחיל מהיום האחרון לפי המועד בו מסתיים ונקבל את עבודה 4, כעת נעבור יום קודם ונבדוק מי המקסימום שנרצה נקבל שזה עבודה 1 ברווח של 40, ולכן מה שנעשה זה שנבצע את עבודה 1 ביום הראשון, ואז נקדים את עבודה 4 ליום השני.

כעת נוכיח את הלמות השונות:

למה 1. למת הבחירה החמדנית, קיים פתרון אופטימלי לבעיית המתכנת, בו נבחר את העבודה עם הרווח המקסימלי ביותר מהסוף בכל פעם.

הוכחה 1: נסמן בOPT את הפתרון האופטימלי עבור העבודה. נסמן j_k את העבודה שמתבצעת בפתרון OPT ביום האחרון (או הכי אחרון) ואת j כעבודה המקסימלית שיכולה להתבצע ביום האחרון. אם $j \in OPT$, אזי בחרנו את הרווח המקסימלי מהסוף. אם $j \notin OPT$, אזי נסתכל על $OPT' = OPT \vee \{j\}$. אם לא קיימת חפיפה בעבודות j וj_k, אזי כיוון ש$|j| > 0$ כיוון שזה סכום של רווח, נקבל כי $|OPT'| = |OPT| + |j| > |OPT|$ ומצאנו פתרון אופטימלי גדול יותר. אם כן קיימת חפיפה בעבודות, בפרט מהנתון שלנו קודם $|j| > |j_k|$ ולכן נסתכל על

$$|OPT'| = |OPT/\{j_k\}| + |j| = |OPT| + |j| - |j_k| \geq |OPT|$$

כלומר בפרט הפתרון הנ"ל יהיה אופטימלי

למה 2. תכונת תת המבנה האופטימלי - בהינתן רשימה כלשהי של עבודות עם רווחים ומועדי ביצוע, אלגוריתם שלוקח את הבחירה החמדנית (הרווח המקסימלי כאשר מתחילים מהסוף), בתוספת פתרון אופטימלי עבור שאר העבודות, הינו אלגוריתם שמוצא פתרון אופטימלי.

הוכחה 2: יהי אלגוריתם A אופטימלי. נב"ש שהטענה לא נכונה ונגיע לסתירה. כלומר, אזי קיים פתרון B כך ש$|B| > |A|$. נסתכל על העבודה האחרונה המקסימלית, נסמנה j_k. בהכרח $j_k \in B$. כמו כן, נראה כי $A/\{j_k\}$ ו$B/\{j_k\}$ הם פתרונות לבעיות ללא העבודה מהיום האחרון (המקסימלית). לפי ההנחה, בפרט $A/\{j_k\}$ הינו פתרון אופטימלי. כלומר,

$$|A/\{j_k\}| \geq |B/\{j_k\}| \Rightarrow |A| - |\{j_k\}| \geq |B| - |\{j_k\}| \Longrightarrow |A| \geq |B|$$

בסתירה לכך שהנחנו B פתרון אופטימלי.

זמן ריצת האלגוריתם החמדן : אם העבודות לא יגיעו ממוינות נצטרך למיין אותם ב$nlogn$, ואח"כ מעבר לינארי על אורך הרשימה בn, סה"כ $O(nlogn)$. עם זאת - נוכיח שניתן למיין ב$O(n)$ עם מיון בסיס.

כיוון שיגיעו מס' עבודות מסויים,נניח n, אזי נצטרך לפחות n ימים לבצע אותם. לכן נעבור על המערך ונוריד את כל מי שמועד הביצוע שלו רחוק מn, כעת יש לנו קלט של $[1, n]$ עבור זמן הביצוע לכן ניתן לבצע מיון בסיס. כיצד? נגדיר $d = log_b n$ כאורך הספרה הארוכה ביותר, אם נקח $b = n$ נקבל $d = log_n n = 1$, כעת נקבל $O(d(n+b)) = O(n+n) = O(n)$.

2024 מועד קיץ א

שאלה 1

סעיף א - נתון מערך בגודל n המכיל ערימת מקסימום בגודל לא ידוע. כלומר x התאים הראשונים של המערך הם ערימה תקנית, שאר התאים מכילים את הערך אנסוף. כאמור, x אינו ידוע. עליכם למצוא את ערכו של x, כלומר האינדקס בו הערימה הופכת ללא תקנית. בזמן $O(logx)$, כלומר אם $n = 2^{100}$, וגודל הערימה בפועל הסתבר כ30, אזי יש לבצע $log30$ פעולות ולא 100 פעולות.

פתרון: הפתרון הנאיבי יהיה ממש לסרוק את המערך מימין לשמאל. זה יעלה לנו $O(x)$. נרצה לשפר, נזכר שראינו בתרגול שבהינתן שאנחנו במערך ממויין, ניתן לבצע קפיצות אקספוננציאליות לחיפוש מספר k ב$O(logk)$. מה מונע מאיתנו להשתמש בזאת גם עכשיו? הפעם אני לא צריך לבדוק האם ממויין או שלא. אני בסה"כ צריך לדעת האם הערך שלי הפך לאנסוף. כלומר נתאר את האלגוריתם הבא: נתחיל באינדקס 1, משם לאינדקס 2,4,8,16 וכן הלאה. בכל שלב אנחנו נבדוק האם הערך של האיבר שאנחנו בו, דהיינו אם אנחנו באינדקס i_s נבדוק את $A[i_s]$ האם הוא אנסוף. אם הוא לא אנסוף - נמשיך הלאה בקפיצות אקספוננציאליות. כעת, בהינתן שקיים שלב $i+1$ בחיפוש שמצאנו שהערך הוא אנסוף, אזי בחיפוש הקודם i הערך לא היה אנסוף. דהיינו, $2^i < x < 2^{i+1}$. כמה זמן יקח לנו לחפש את x במקטע הזה? בדיוק $logx$, וכן מס' הקפיצות שעשינו עד שהגענו לשלב זה גם כן היה $logx$, לכן סה"כ ביצענו $2logx$ עבודה, כלומר $O(logx)$ זמן. מש"ל.

סעיף ב - נתונה ערימת מינימום בה בכל רמה כל האיברים זהים, כלומר הערימה $A = [1, 2, 2, 5, 5, 5, 5]$ מקיימת זאת. כתוב אלגוריתם שמקבל כקלט ערימה כנ"ל ואיבר נוסף z, ומחזירה אינדקס כלשהו של z בערימה או -1 אם z לא נמצא. זמן הריצה צריך להיות $O(log(logn))$ כאשר n הוא מס' האיברים בערימה

פתרון: ננסה לנתח מה הסיבוכיות אומרת בשביל לנסות לגשת לפתרון. אם נסמן כקלט $k = logn$, נרצה לבצע את הפתרון ב$O(log(k))$, כלומר אם נצמצם את החיפוש שלנו לחיפוש על $logn$ איברים, החיפוש יהפוך להיות חיפוש רגיל בערימה. נבנה מערך עזר B וננסה להשתמש בשתי העובדות הבאות:

1. זו ערימת מינימום, כיוון שכל הרמות מכילות את אותם איברים בכל רמה, בפרט מערך A הינו ממויין (בשל העובדה שאב צריך להיות קטן מהבן שלו בערימה).

2. זה עץ בינארי כמעט שלם.

אנחנו בעצם רוצים לבצע חיפוש בינארי, על $log(n)$ איברים. ובכן, אנחנו יודעים שגובה של ערימה הינו $O(log(n))$. כלומר, ישנם $logn$ איברים שונים בדיוק במערך הנ"ל. אם נתחיל בחיפוש בינארי, מהאיבר האמצעי וכך נבדוק כל פעם, נקבל סיבוכיות שהיא $O(logn)$. לכן ננצל את העובדה שאנחנו לא צריכים לעבור על כל האיברים בתוך המקטעים הקטנים עוד יותר.

כלומר, נרצה לבצע חיפוש בינארי רק על $logn$ האיברים השונים. מתכונת הערימה המיוחדת, נראה כי האיברים השונים מופיעים בדיוק במיקומים האקספוננציאלים המתאימים לחזקת 2. כלומר, האיברים ה"שונים" הם באינדקסים $1,2,4,8,16....$ מה שנעשה יהיה לבצע חיפוש בינארי, על הערכים האקספוננציאלים בחזקת 2 . וקצת יותר פורמלי:

אם יש לנו n איברים, אזי יש $logn$ איברים שונים באינדקסים $1,2,4,....2^{log_2 n} = n$ (נעיר כי אכן ייתכן שלא מדובר בעץ שלם, שכן ערימה היא עץ כמעט שלם, אבל מובטח לי ברמה האחרונה איבר אחד לפחות וזה מספיק לי), יש כאן סה"כ $logn$ איברים, ונבצע כעת חיפוש בינארי על הערכים האקספוננציאלים הללו, סה"כ זה יעלה $O(log(logn))$.

שאלה 2

25 נקודות. נניח שיש לנו מבנה נתונים של מחסנית שתומכת בפעולות הבאות:

1. דחיפה - מכניס את האיבר x למחסנית
2. הוצאה - מוציא את האיבר העליון מהמחסנית
3. ביטול: בהינתן k (לא ידוע מראש), החזר את המחסנית למצב שהייתה לפני k פעולות דחיפה/הוצאה אחרונות - כלומר מבצעים ביטול של k פעולות $push/pop$ האחרונות. ניתן להניח שאם הפעולה נקראת עם k, אזי בוצעו לפני כבר k פעולות.

הציעו מימוש למבנה כך שהעלות לשיעורין של כל פעולה תהיה $O(1)$, פרט ניתוח כל פעולה במקרה הגרוע ביותר.

פתרון: נרצה להחזיק מחסנית עזר. תהי מחסנית S. נשתמש במחסנית עזר S' כאשר S' תכיל 2 שדות. בשדה הראשון יהיה ביט, כאשר נקבע כלל: 1 מסמן הכנסה, 0 מסמן הוצאה. והשדה השני יהיה מספר. שהמספר זה הערך שהכנסנו או הוצאנו. כעת נתאר את מימוש הפעולות הבאות, ולבסוף ננתח לשיעורין בעזרת שיטת הבנק כל אחת מהפעולות + המקרה הגרוע.

א. דחיפה: נכניס למחסנית S את הערך x, וכן נכניס למחסנית S' את השדות $(1,x)$ כאשר שוב 1 יציין הכנסה.

ב. הוצאה: נוציא מהמחסנית S את הערך x, ונכניס למחסנית S' את השדות $(0,x)$ כאשר שוב 0 יציין הוצאה.

ג. ביטול: בהינתן שקיבלנו k, נבצע את התהליך הבא: נשלוף k פעמים מהמחסנית S' את הערך שבראש המחסנית. אם שלפנו ערך ששדה הביט שלו הינו 0 אזי נבצע popמהמבנה. אם שלפנו ערך ששדה הביט שלו הינו 1 אזי נעשה $push$ של x למבנה. לאחר k פעולות המבנה ישאר כשהיה.

כעת ננתח לשיעורין:

ננתח באמצעות שיטת הבנק. נפזר מטבעות.

א. $push$- נתן לכל פעולה 4 מטבעות. מטבע ראשון יהיה מנוצל באופן מיידי, כאשר נבצע למבנה $push$ על הערך. מטבע שני יהיה להכיס את הפעולה $(1,x)$ לתוך S' 2 מטבעות ינוצלו לשימוש עתידי, כאשר אם נבצע את הפעולה ביטול על k איברים, במידה ונתקל בפעולת $push$ שם, נצטרך מטבע אחד על מנת לבצע pop לערך $(1,x)$ מהמבנה S' ועוד מטבע על מנת לבצע $push$ של x לתוך המבנה בחזרה.

ב. pop- בדומה, 4 מטבעות לכל פעולה. אחד ינוצל מיידית,מטבע שני יהיה להכיס את הפעולה $(0,x)$ לתוך S', 2 לשימוש עתידי כאשר הראשון ישומש כאשר נוציא את $(0,x)$ מהמחסנית S' והשני ישומש כאשר נבצע את הפעולה pop של x מהמבנה.

ג. ביטול - כעת לא ניתן מטבעות כלל, נראה כי הפעולות pop ו$push$ שילמו על העלות של הפעולה, ובהינתן k פעולות כאלו זה יבוצע ע"י שימוש במטבעות שלהם.

נשים לב שהפעולה ביטול לא יכולה לבוא כל הזמן, כיוון שבהינתן שאנחנו רוצים לקרוא לה עם ערך של k, אזי בוודאות התבצעו k פעולות קודם לכן. שילמנו כבר על עלות הפעולה ביטול, לכן היא עולה 0 מטבעות.

סה"כ כל אחת מהפעולות חסומה ב4 מטבעות, ולכן לשיעורין כל אחת מהפעולות עולה $O(1)$ בדיוק.

שאלה 3

הציעו מבנה נתונים S אשר מתחזק n מספרים שלמים שונים זה מזה ותומך בשגרות הבאות בזמני הריצה המצויינים:

א. $Insert(S,k)$ הכנסת המספר השלם k למבנה S בזמן $O(logn)$

ב. $Delete(S,k)$ מחיקת המס k מהמבנה S בזמן $O(logn)$
ג. $SwitchSign(S,k)$ החלפת הסימן של המס' k בS אם הוא קים. $O(logn)$
ד. $SearchNonNegetive(S,k)$ מחזיר את המפתח האי שלילי המינימלי בS שגדול או שווה למס' השלם k . k לא בהכרח מס' ששמור בתוך S. זמן ריצה $O(logn)$
יש לתאר אלגוריתמים באופן קצר ובמדויק. נתח בקצרה סיבוכיות.

פתרון: אם נבחן את השאלה לעומק, נראה שאין בידינו ברירה אלא להשתמש בעץ.
, לכן אנחנו מחפשים מבנה שתומך "בתכלס" בהכנסה הוצאה ומחיקה ב$O(logn)$, ואיזה יופי למדנו עצי AVL שעונים על כל זה מבלי שנעבוד כעת בלהסביר הכנסה והוצאה.
הפעולה השלישית די שקולה לחיפוש בעץ מאוזן, ובכן אחרי שנמצא אותו (אם נמצא אותו) מה שיעלה לנו $O(logn)$, נוציא אותו מהעץ $O(logn)$ ונכניס אותו שוב לעץ עם ערכו הנגדי , וכן הכנסה עולה $O(logn)$ לכן סה"כ ביצענו $3logn = O(logn)$.
כעת, נותר להסביר את הפעולה האחרונה.
נרצה לבצע כעת מס' שינויים במבנה, נראה שדי נתקענו.
נחליט שנשמור שני עצים, אחד לערכים חיובים ואחד לשלייליים.
בהינתן שיגיע ערך חיובי נכניס אותו לעץ החיובי, ובהינתן שיגיע ערך שלילי נכניס אותו לעץ השלישי. הכנסה והוצאה לעץ AVL אכן עולות $O(logn)$.
באשר לפעולה השלישית - כמו שתיארנו מעלה, אם הוא חיובי בלי הגבלת הכלליות נחפש אותו בעץ החיובי $O(logn)$, נוציא אותו $O(logn)$ ונכניס אותו לעץ השלילי $O(logn)$, סה"כ $3logn = O(logn)$
כעת לפעולה האחרונה. אם k שלילי, אזי נחזיר את איבר המינימום בעץ החיובי. זה יעלה $O(logn)$.
אחרת, אם k קיים, היחיד שמינימלי ממנו וגדול שווה-לו זה הוא בעצמו. כלומר נחפש ונחזיר אותו $O(logn)$. אחרת, הוא לא קיים. נחפש את העוקב של k בעץ. אנחנו יודעים שהוא לא קיים לכן אנחנו נגיע במסלול שלו לעלה כלשהו. כעת, לאחר שהגענו לעלה נעלה מעלה במסלול החיפוש עד שנמצא את זה שגדול-שווה לו מינימלית (כל פעם נשווה ונבדוק) וזה יעלה כגובה העץ $O(logn)$.

שאלה 4

סעיף א - 13 נקודות. נתון מערך A בגודל n של מספרים כלשהם. ניתן להניח שהם שונים. ברצוננו לבנות מערך S בגודל n כך שבכל אינדקס i במערך S יהיה את כמות האיברים שנמצאים משמאל לi וקטנים או שווים מ$A[i]$ במערך המקורי.
הראו שכל אלגוריתם לביצוע המשימה יקח לפחות $\Omega(nlogn)$ פעולות.

פתרון: כאשר נותנים לי נתון כזה כללי, ישר חושבים על מיון מבוסס השוואות. בואו ננסה בהינתן הנתונים הנוכחיים, לבצע מיון. כלומר, נניח שקיים אלגוריתם כזה שמעביר את הנתונים. זמן הריצה של האלגוריתם יהיה T. כעת, ננסה להראות שבהינתן ביצוע עבודה מקדימה ב$O(T)$, נוכל לקבוע לכל איבר את המיקום שלו ב$O(1)$, מה שיגרור שזמן הריצה הכולל יהיה

$$T + n * O(1) = O(T + n)$$

ולכן כיוון שקיבלנו מיון, בפרט יתקיים כי $T + n \geq nlogn$, כלומר $T \geq nlogn - n$, לכן נוכיח ש$T \in \Omega(nlogn)$.
כעת, לאחר ההסבר המתמטי, כל שנותר הוא להוכיח שניתן בכלל לבצע זאת.
כעת אני נמצא בתא i במערך, מה זה עוזר לי? בהינתן $A[i]$, אם אני מסתכל על $S[i]$ אני יודע כמה איברים משמאלי קטנים מ$A[i]$.
נסתכל על מערך $B[i]$ שיתאר כמה איברים קטנים מימין (קל לחשב כי $B[i] = A[n-1-i]$
כעת נשים לב לדבר הבא, $S[i] + B[i]$ הוא האיבר הi במערך הממויין
מדוע? כל איבר יכול להיות או מימין או משמאל לאיבר כלשהו, לכן יספר בדיוק פעם אחת ברשימת כל האיברים שקטנים מהמחובר הi בA. כעת מסדרים את האיברים לפי ההגיון הזה, זה יעלה $O(n)$, נקבל סתירה כפי שרצנו בתוספת T, ולכן $T = \Omega(nlogn)$.
סעיף ב - תארו אלגוריתם לביצוע המשימה שרץ בזמן אופטימלי לבעיה.
פתרון: ראינו שהזמן האופטימלי, הולך להיות $O(nlogn)$. אני רואה $logn$, נרצה להשתמש בעץ AVL. אנחנו רוצים לבנות מערך S שכזה. נכניס את האיבר הראשון לעץ, תמיד נעדכן את $S[0]$ להיות אפס, שהרי אף אחד משמאל לא קטן ממנו כי אין אף אחד משמאלו. כעת, נעבור לאיבר הבא (השני) במערך ונכניס אותו לעץ. כעת נשים לב, מה אנחנו רוצים למצוא? אני רוצה שבהינתן שאני מכניס ערך לעץ, כלומר אני מגיע לתוך העץ, לבדוק כמה איברים משמאלי קטנים ממני. מי נמצא משמאלי? כל מי שהכנסתי לעץ. לכן, השאלה כעת תומר להיות "כמה איברים יש בעץ שקטנים ממני?", ובכן לסרוק עץ עולה $O(n)$ ואנחנו לא מעוניינים לסרוק עץ כי אז נגיע לסיבוכיות

$O(n^2)$. מה כן? בהינתן שאנחנו בתוך עץ חיפוש, נוכל לבצע זאת ב$O(logn)$, מיד אפרט יותר על כך, בכל מקרה, נניח כרגע שניתן לבצע זאת ב$O(logn)$, אנחנו בכל שלב נקח את האיבר, ונשאל כמה איברים בעץ קטנים ממנו, שכן האיברים בעץ זה האיברים משמאל, ובהתאם לכך נעדכן את הערך בתוך S. כך ביצענו n הכנסות, נראה כי בכל הכנסה ביצענו 2 פעולות : הכנסה לעץ שעלתה $O(logn)$ וכן שאילתא "מי קטן ממני" שעלתה $O(logn)$. סה"כ בוצעו $O(n(logn + logn)) = O(2nlogn) = O(nlogn)$

כעת נסביר מדוע ניתן בתוך עץ AVL לדעת בהינתן מס' כמה קטנים ממנו בתוך העץ שנכנס. נוסיף לכל איבר בעץ שתי שדות, אחד יהיה $size_R$ והשני יהיה $size_L$ כאשר הם יתארו את גודל תת העץ. ראינו כבר בתרגול שהוספת השדות לא תשנה הכנסה והוצאה לAVL מבחינה אסימפטוטית שכן השינויים שידרשו לעשות הם מס' קבוע של שינויים. כעת, בהינתן שאני מקבל ערך k, נכניס אותו לתוך העץ, ולאחר מכן נבדוק את הגודל $size_L$. מדוע? למעשה הכנסנו את האיבר לתוך העץ, כעת הוא בעץ עם כל האיברים שהיו משמאלו. הוא נמצא במיקום כלשהו בעץ, כפי שאמרנו זה עץ חיפוש לכן כל מי שמשמאלו יהיה קטן ממנו, וכל מי שמימינו גדול ממנו. לכן כל שנדרש הוא לקחת את $size_L$ שלו וזה יהיה הערך. ושוב, כפי שכבר הבהרתי מעלה, אסימפטוטית סיבוכיות הפתרון תהיה $O(nlogn)$, כנדרש.

2012 - שאלה 2 - מועד ב' טומי

הצע מבנה נתונים S שתומך בפעולות הבאות ובדרישות הסיבוכיות הבאות:

א. $Build(S, L)$ - בניית המבנה S מאיברי הרשימה של נתוני הקלט L בזמן $O(n)$

ב. $Max(S)$- החזרת ערך מקסימלי בזמן $O(1)$

ג. $Min(s)$= החזרת הערך המינימלי בזמן $O(1)$

ד. $Delete - max(S)$ מחיקת האיבר המקסימלי בזמן $O(logn)$

ה. $Delete - Min(s)$ מחיקת האיבר המינימלי בזמן $O(logn)$

פתרון: נבנה ערימת "מינימום מקסימום", כלומר נתחזק שני ערימות. הערימה הראשונה תהיה ערימת מקסימום ותקרא S_1 והערימה השנייה תקרא S_2 ותהיה ערימת מינימום. נבנה את שני הערימות יחד, ונכניס את איברי הקלט אל כל אחת מהן, בניית ערימה עולה $O(n)$ ולכן בניית שתי ערימות תעלה $2n = O(n)$. כמו כן, בבנייה אנחנו נקשר את האיברים באמצעות מצביעים. זה לא ישנה את הסיבוכיות האסימפטוטית שתשאר לינארית. נתקדם הלאה, כאשר נרצה להחזיר את הערך המקסימלי נחזיר את הערך שבראש הערימה s_1, וכאשר נרצה את המינימלי נחזיר את זה שבראש הערימה S_2, סה"כ כל אחת מהפעולות תעלה $O(1)$. כעת, ניגש למחיקת האיבר המקסימלי. נלך אל הערימה S_1 ונרצה למחוק את איבר המקסימום, לפני כן ניגש עם המצביע שלו לאיבר המקביל שלו בערימת S_2, נמחק אותו, מחיקה מערימה עולה $O(logn)$ בהינתן שאנחנו יודעים את הערך שברצוננו למחוק (מדובר בהחלפות ורוטציות כגובה העץ וכן יש לנו מצביע אל האיבר!!!!), ואז כמובן שנמחק את הראש בS_1 שכן ערימת מקסימום ולמחוק את הערך שבראשה עולה $O(logn)$, באופן סימטרי ממש עם מחיקת המינימלי רק שנלך לראש S_2 ועם המצביע נגיע אל S_1 לאותו איבר, נמחק מכל ערימה את האיבר יעלה $2logn = O(logn)$. מש"ל.

מבחן 2010 טומי קליין מועד א

שאלה 1: נתבונן בשיטה הבאה למיון קבוצה A_0 של n מספרים שונים. נחלק את A_0 ל$\sqrt{n}$ תתי קבוצות זרות עם $\sqrt{n}$ איברים בכל אחת. לאחר מכן נבחר בכל פעם את האיבר הקטן ביותר מתוך אלו שעדיין לא נבחרו ע"י זה שנמצא בזמן לינארי את האיבר הקטן ביותר בכל אחת מ$\sqrt{n}$ הקבוצות (נקרא לקבוצת $\sqrt{n}$ האיברים הקטנים האלו A_1,) ואז נמצא את האיבר הקטן ביותר בA_1.

א. כמה זמן לוקח למצוא את האיבר הקטן ביותר? והשני? והn-יי? סכם - מה הסיבוכיות של שיטת מיון זאת?

פתרון: למעשה מחלקים את האיברים ל$\sqrt{n}$ קבוצות, כעת אנחנו נרצה בזמן לינארי בכל קבוצה למצוא את האיבר הקטן ביותר. בכל קבוצה יש $\sqrt{n}$ איברים ולכן לפי השוואה אנחנו נזדקק ל$O(\sqrt{n})$ פעולות על מנת למצוא את האיבר הקטן ביותר בקבוצה בודדת. עבור כל הקבוצות יעלה $O(n)$ למציאת המינימומים. כעת נרצה לחפש את המינימום מבין שורש אן המינימומים ולכן זה יעלה $O(\sqrt{n})$, סה"כ $n + \sqrt{n} = O(n)$ יעלה לדעת את הקטן ביותר (או השני והשלישי... שקול מחשבתית) במשך n פעמים ולכן סה"כ $O(n^2)$.

ב. האם יעזור אם במקום לחזור ולחפש שוב ושוב את האיבר הקטן ביותר בקבוצה, אם נמיין את $\sqrt{n}$הקבוצות מיד בהתחלה?

פתרון: נבדוק האם זה יעזור. נניח ונחליט שבכל קבוצה נבצע מיון, ידוע כי מיון הוא עם חסם תחתון של $|X|log|X|$ ולכן במקרה שלנו מיון בתוך קבוצה יעלה $(n^{0.5}logn^{0.5}) = \frac{1}{2}\sqrt{n}logn = O(\sqrt{n}logn)$ למיון קבוצה ספציפית. על התהליך נצטרך לחזור $\sqrt{n}$פעמים ולכן מיון כל הקבוצות הקטנות יעלה $O(nlogn)$. כעת, בכל שלב נצטרך לקחת את $\sqrt{n}$ האיברים הקטנים ביותר (הראשונים בכל אחת מהקבוצות) ולבדוק מי הכי קטן מבניהם. זה יעלה $O(\sqrt{n})$. נעיר כעת כי להחליט שניקח תמיד את הראשונים בכל קבוצה ונמיין אותם וככה נבנה מיון זה לא נכון כי יתכן שיש קבוצה שכל האיברים בה גדולים מאוד ולכן הקטן מתוכם יהיה עדיין גדול מאוד יחסית ביחס לאיברים אחרים. לכן מה שנעשה יהיה לקחת בכל פעם את האיבר הראשון בכל קבוצה וסה"כ נקבל $\sqrt{n}$ איברים למיין אותם מה שיעלה $O(\sqrt{n})$, על אותו התהליך נרצה לחזור n פעמים ונקבל שלמיין יעלה $O(n^{1.5})$, סה"כ $nlogn + n^{1.5} = O(n^{1.5})$. אכן עזר.

ג. כדי לשפר נרצה לחלק את A_0 כעת ל$n^{\frac{2}{3}}$ קבוצות בגודל $n^{\frac{1}{3}}$ כל אחת, לאחר שנבצע את התהליך בו נבחר את המינימומים בכל אחת נקבל קבוצה A_1 ובה $n^{\frac{2}{3}}$ איברים. לכן גם אותה נחלק ל$n^{\frac{1}{3}}$ קבוצות בגודל $n^{\frac{1}{3}}$ כל אחת. נקרא לקבוצת המינימליים כעת, A_2. כמה זמן כעת לוקח למצוא את האיבר הקטן ביותר? והשני? סכם - מה סיבוכיות שיטת מיון זו?

פתרון: כעת אנחנו בוחרים את המינימומים בכל קבוצה מה שיעלה $O(n^{\frac{1}{3}})$ ונעשה זאת על $n^{\frac{2}{3}}$קבוצות ולכן עלה לנו $O(n)$ לקבל את הקבוצה A_1. כעת נרצה לחזור על התהליך ולחפש בכל קבוצה את המינימום מתוכה, קיבלנו סה"כ שיש $n^{\frac{1}{3}}$ קבוצות, בגודל $n^{\frac{1}{3}}$ כל אחת. למצוא מינימום באחת יעלה $O(n^{\frac{1}{3}})$ ולכן עבור כולן יעלה $O(n^{\frac{2}{3}})$. סה"כ זה כמה שעלה למצוא את A_2. כעת, קיבלנו קבוצה בגודל $n^{\frac{1}{3}}$, למצוא מינימום בתוכה יעלה $O(n^{\frac{1}{3}})$ ולכן אם נסכם את התהליך עלה למצוא את המינימום $O(n+n^{\frac{2}{3}}+n^{\frac{1}{3}})=O(n)$. סה"כ את התהליך נרצה לעשות n פעמים ושוב לצערנו נקבל $O(n^2)$

ד. תאר הכללה של סעיף ג' לשתי רמות נוספות והכלל עבור k רמות (שים לב שבכל קבוצה צריכים להיות לפחות 2 איברים)

פתרון: מסימטריות התהליך נקבל למעשה $O(n+n^{\frac{4}{5}}+n^{\frac{3}{5}}+n^{\frac{2}{5}}+n^{\frac{1}{5}})=O(n)$ ולכן לn איברים יעלה $O(n^2)$. עבור k רמות נשים לב שכל קבוצה חייב להיות 2 איברים ולכן $max_k=\frac{n}{2}$ ולכן כעת למצוא מינימום יעלה $O(\sum_{i=1}^{k} n^{\frac{1}{k}})=O(n)$ וכן את כל האיברים שוב $O(n^2)$.

שאלה 5:

נתון מערך של n איברים בשם A כך שk האיברים הראשונים במערך הם מספרים זוגיים ושאר האיברים הם מספרים אי זוגיים. k אינו ידוע. תאר אלגוריתם למציאת k בזמנים הבאים:

א. $O(logn)$

ב. $O(min(logn,k))$

ג. $O(logk)$

פתרון: נוכל להתייחס לנתון כאילו אנחנו יודעים שהמערך "ממויין" אבל לא באמת. כלומר, לסעיף הראשון נרצה לעשות חיפוש בינארי. נלך תמיד לאיבר הראשון. אם הוא זוגי, אזי נלך לחצי הימני כי השבירה של k מתבצעת בצד ימין. אם הוא אי זוגי, אזי השבירה התבצעה כבר ונלך לצד השמאלי. סה"כ נבצע $T(n)=T(\frac{n}{2})+O(1)=O(logn)$.

באשר לפתרון השני נרוץ במקביל מההתחלה כאשר נבדוק כל איבר משמאל לימין אם הוא זוגי או שלא, ונפעיל את האלגוריתם מסעיף א' תוך כדי. נפסיק כאשר נמצא את k, כלומר נפסיק כאשר זמן מינימום של $O(min(logn,k))$

לפתרון השלישי נרוץ בקפיצות אקספוננציאליות. נתחיל מ1,2,4,8 ובדומה. נניח שהגענו לאינדקס i עבורו האיבר אי זוגי, אזי באינדקס הקודם 2^{i-1} האיבר היה אי זוגי, סה"כ ביצענו $logk$ קפיצות עד לכאן וכעת נותר לחפש בתוך התחום $2^{i-1}<k<2^i$. נראה כי $k<2^i$ גורר $i<logk$ ולכן $i=O(logk)$ כלומר זו הצדקה לטענה מתחילת השורה שביצענו "$logk$ קפיצות" וכן כעת התחום הוא של $2^i-2^{i-1}=2^i-\frac{1}{2}2^i=2^{i-1}$ איברים, כמו כן ניתן לעשות חיפוש בינארי בתוך התחום שיעלה $O(log_2 2^{i-1})=O(i-1)=O(logk)$ וסה"כ ביצענו $2logk=O(logk)$.

שאלה 6:

האלגוריתם סלקט שנלמד בהרצאה מקבל כקלט מערך לא ממויין בגודל n ומספר i בטווח $[1,n]$ ומחזיר את האיבר הi בגודלו בזמן $O(n)$. באלגוריתם חילקנו את המערך ל$\frac{n}{5}$ קבוצות בנות 5 איברים בכל קבוצה. כעת נבצע שינוי באלגוריתם - נניח שהיינו מחלקים את המערך ל$\frac{n}{3}$ קבוצות שבכל קבוצה יש 3 איברים.

א. כמה איברים לפחות מצליח לנפות חציון החציונים?

ב. כתוב נוסחת נסיגה לאלגוריתם ופתור אותה. הוכח הפרך: עדיף אסימפטוטית להשתמש בחלוקה ל$\frac{n}{3}$ קבוצות.

פתרון:

א. ננסה להשוות את התהליך לתהליך שהתבצע בסלקט. מצאנו את החציונים, ומתוכם מצאנו את חציון החציונים. גילינו שלאחר שסידרנו זאת, בחצי מהאיברים שקטנים מחציון החציונים 3 איברים לא רלוונטים עוד - החציון ואלו שמשמאלו, ולכן נפטרנו מ$\frac{1}{2}*\frac{3n}{5}=\frac{3n}{10}$ מהאיברים. כעת כאן, בכל קבוצה מהחצי האלו אנחנו נפטר מאיבר שמשמאל לחציון והאיבר האמצעי, כלומר $\frac{1}{2}*\frac{2n}{3}=\frac{n}{3}$ מהאיברים. כלומר לאחר כל תהליך נקבל שנפטרנו מ$\frac{n}{3}$ מהאיברים שקטנים מחציון החציונים ויש $\frac{2n}{3}$ איברים שגדולים מחציון החציונים. לכן סה"כ נקבל את נוסחת הנסיגה $T(n)=T(\frac{2n}{3})+T(\frac{n}{3})$ שכן חילקנו לקבוצות של 3 וכן הראינו מאיפה הגיע השני שליש כעת. עכשיו נראה כי

$$T(n)=T(\frac{2n}{3})+T(\frac{n}{3})+O(n)$$

נוכיח כעת כי $T(n)\in O(nlogn)$. באינדוקציה בסיס על $n=1$ כמובן טריוואלי, נניח נכונות עבור קלטים עד גודל n כלומר $T(n')\leq cn'logn'$. נוכיח נכונות עבור n ונשתמש בהנחת האינדוקציה:

$$T(n)=T(\frac{2n}{3})+T(\frac{n}{3})+O(n)\leq\frac{2cnlog(\frac{2n}{3})}{3}+\frac{cnlog(\frac{n}{3})}{3}+n=$$

$$\frac{2cn(log2n - log3) + cn(logn - log3) + 3n}{3} =$$

$$\frac{2cnlog2 * logn - 2cnlog3 + 2clogn - cnlog3 + 3n}{3}$$

$$= \frac{logn(2cnlog2 + 2c) - 3cnlogn + 3n}{3} \le \frac{c * 2log2 * nlogn + 2clogn + 3n}{3}$$

$$\le \frac{c * 2log2 * nlogn + 2cnlogn + 3nlogn}{3} =$$

$$nlogn(\frac{2clog2 + 2c + 3}{3}) = O(nlogn)$$

סה"כ הטענה לא נכונה ולא כדאי אסימפטוטית להשתמש בחלוקה לשלוש. מש"ל.

2018 **מועד ב' טופס שאלה 2**

נתונים m מערכים לא ממויינים $A_1, ...A_m$ שאיבריהם כולם מספרים שלימים בתחום $[1, k]$, יהי n המספר הכולל של איברים בכל המערכים יחד. כלומר $n = |A_1| + ... + |A_m|$. כתוב אלגוריתם שממיין את כל m המערכים ביחד, בזמן כולל של $O(k+n)$ במקרה הגרוע. כלומר, לאחר הרצת האלגוריתם כל אחד מהמערכים יכיל את אותם איברים שהכיל לפני ההרצה אך כעת הם יהיו ממויינים מהקטן לגדול.

פתרון: אנחנו יודעים שהתחום הוא $[1, k]$ וכן נסמן גודל קבוצה i כX_i. אזי, ניתן להשתמש במיון בסיס למיין כל קבוצה, ע"י בחירת בסיס k ולקבל שאורך המילה הכי גדולה הוא $d = log_k k = 1$ וכן הבסיס $R = k$ ולכן $O(d(x_i + k)) = O(k + x_i)$, כלומר זו העלות למיון מערך יחיד. כמה יעלה למיין את כל המערכים? אם נסכום זאת, נקבל

$$\sum_{i=1}^{m} k + x_i = mk + n$$

ובכן, זו לא דרישת הסיבוכיות שביקשו. לכן אנחנו נצטרך לשפר את הרעיון: נראה כי אם נקח את כל האיברים ונוסיף להם "תווית" שתציין מאיזה קבוצה הם, נוכל למיין את כל n האיברים באמצעות אותה שיטה שכן גם הם בתחום $[1, k]$. נבחר בסיס k ונקבל אורך מילה $d = log_k k = 1$ וכן בסיס $R = k$ ולכן המיון הכולל יעלה $O(1(k+n)) = O(k+n)$, כעת, נעבור פעם נוספת על האיברים ונשלח כל איבר לקבוצה שלו לפי המזהה שהדבקנו ולכן זה עוד n סה"כ $k + n + n = O(k + n)$. הערה - נעבור כמובן לפי סדר המיון שנשלח אותם חזרה.

2018 מועד ב' טופס שאלה 4

ברצוננו להדפיס את m האיברים הגדולים ביותר מתוך קבוצה של n איברים המאוחסנת בערימת מקסימום. כאשר כמובן $0 < m < n$ וכן $O(logm) \neq O(logn)$, הראה כיצד לעשות זאת בזמן $O(mlogm)$. רמז: השתמש בערימת עזר.

פתרון: נבנה ערימת מקסימום חדשה. נסתכל על השורש ונדפיס אותו. נוציא אותו. כעת נכניס לערימת העזר את הבנים של השורש (באמצעות פוינטרים ככה נדע בכל שלב את מי להכניס). נחזור על התהליך שוב ושוב עד שנסיים ונגיע לm. בכל שלב גובה הערימה החדשה יהיה חסום ב$logm$ ונבצע סה"כ הכנסה והוצאה כל פעם כמס' פעולות קבוע במשך m פעמים ולכן $O(mlogm)$. מש"ל.

2019 שאלה 5

בפרלמנט של מדינת אלגו־לנד, ישנם n מושבים. הפרלמנט מורכב מk מפלגות שונות. נסמן את מספר המושבים בפרלמנט של המפלגה i בs_i כך ש$n = s_1 + s_2 + ... + s_k$. קואליציה בפרלמנט חייבת לכלול יותר מחצי מהמושבים. הקואליציה מורכבת ממפלגות (כל חברי מפלגה מסוימת נמצאים בקואליציה או שכולם נמצאים מחוץ לקואליציה).

לכל חבר כנסת בקואליציה משלמים משכורת גבוהה במיוחד, ולכן מטרת המדינה היא להרכיב קואליציה בגודל קטן ביותר. אך כמובן שהיא חייבת לכלול יותר מחצי ממושבי הפרלמנט.

תאר אלגוריתם תכנון דינמי שמקבל את גדלי המפלגות $s_1,....,s_k$. הפלט יהיה גודל הקואליציה הקטן ביותר שאפשר להרכיב ממפלגות בגדלים אלו. תארו סיבוכיות זמן ומקום.

פתרון: נרצה להסתכל על שני מקרים. עבור כל מפלגה ישנם 2 אפשרויות בדיוק: נוכל להכניסה לקואליציה או שלא. נגדיר את הפונקציה הבאה $f(i,x)$ שתחשב את גודל הקואליציה כאשר עברנו על המפלגות $1,...i$ בגודל x כאשר נאתחל את x להיות $\frac{n}{2}+1$ או $\frac{n}{2}$ בערך שלם עליון ־ כלומר בסף הדרוש, ופתרון יתקבל כאשר $x \leq 0$ שכן זה אומר שניתן להפסיק להוסיף מפלגות. נראה כי $1 \leq i \leq k$ וכן לא יתכן $i > k$. אם $i > k$ וגם $x \leq 0$ אזי נגדיר את הערך להיות אפס ואם $i > k$ וגם $x > 0$ אזי עברנו על מס' מפלגות גדול מדי שבו לא מצאנו פתרון ולכן נשים אנסוף. אחרת, נתעדף אופטימזיציה. נתאר את נוסחת הנסיגה הבאה:

$$f(i,x) = \left\{ \begin{array}{cc} \infty & i > k \wedge x > 0 \\ 0 & i > k \wedge x \leq 0 \\ min\{f(i-1,x), s_i + f(i-1, x-s_i) & o.w \end{array} \right\}$$

כאשר אנחנו בוחרים את האפשרות המינימלית מבין לקחת את המפלגות עד $1,...i-1$ ולא לקחת את הערך לבין עם לקחת.

נבנה מטריצה בגודל $k+1 \times n+1$ ונמלא אותה לפי המתואר מעלה. בהינתן שאני בתא מסויים, יתכן שאני זקוק לתאים משמאלי או מעליי, ולכן אם נמלא משמאל לימין לפי השורות זה ימלא לנו כל תא ב$O(1)$, כעת הפתרון לבעיה יכול להמצא בכל אחד מהתאים שיקיימו $x \leq 0$ ולכן נסרוק שוב. סה"כ עלות סיבוכיות זמן ומקום $O(nk)$.

מבחן 2022 מועד ב' קיץ שאלה 2

א. הוכח הפרך ־ יהי A מערך עם n מספרים טבעיים בטווח $[1,k]$ כך ש$k = O(n)$. אזי קיים מבנה נתונים שזמן בנייתו לינארית ומאפשר ב$O(1)$ לענות על השאילתה הבאה ־ בהינתן a,b מס' טבעיים החזר את כמות המספרים בA שנמצאים בטווח בין a,b כולל.

פתרון: הוכחה ־ נבנה את המבנה הבא ־ ראשית נבנה מערך ונמיין אותו באמצעות מיון בסיס, כאשר הבסיס יהיה n ואז $d = log_n n = 1$ ונקבל $O(1(n+n)) = O(n)$ לבניית המערך הממויין. כעת אנחנו יודעים כי מס' האיברים הוא עד k והוא ב$O(n)$. נבנה טבלת האש בגודל n וכל איבר בטבלה יהיה איבר מהמערך הממויין. בעצם נרצה לבנות מעין מערך או מבנה שבכל רגע נתון, יגיד לי כמה איברים יש לי שקטנים שווים מi כאשר $1 \leq i \leq k$. זו תהיה מעין פונקציה f. ואז, בהינתן שנקבל שני מספרים נחזיר את $f(b) - f(a)$ וזה אכן יהיה מס' המספרים בA שנמצאים בטווח וזה יעלה $O(1)$. כעת נותר לבנות את המבנה הזה באמצעות טבלת האש. נבנה את הטבלה כך: נעבור על k האיברים שלנו ונחזיק אותם במערך בגודל k בשם B. כעת עבור כל איבר k נחפש האם הוא קיים בתוך טבלת האש שלנו. כלומר, נתחיל מ$k = 1$. חיפשנו ומצאנו שהוא קיים, אזי נעדכן את האינדקס במערך החדש הזה B ונגדיל את מס' האיברים. אם מצאנו שקיים אנחנו נוציא אותו מהטבלה, ונמשיך לבדוק כי יתכן שיש מס' שחוזר על עצמו פעמיים. נפסיק לבדוק כאשר נקבל שלא קיים איבר כזה במבנה. ואז, נעבור לאיבר 2. כעת, אנחנו נגדיר $B[2] = B[1] + ...$ כאשר הפלוס יהיה האיברים שאנחנו כעת נוסיף אם נמצא. למעשה בכל שלב אנחנו בונים רקורסיבית מהתא הקודם + בודקים כמה איברים שווים למס' עצמו k. נשים לב כי בשיטה זו, אם איבר קטן שווה מ1, הוא בפרט קטן שווה מ2 וכדומה. כלומר אם נסכם פורמלית: בהינתן שאני באינדקס i, נגדיר $B[i] = B[i-1] + sumOfOuters$ שכן ההוצאות יהיו מס' האיברים שנוציא מהטבלה בשיטה שתוארה קודם. אם טבלת ההאש תתרוקן, אזי שמצאנו את הערך האחרון והוצאנו אותו. נניח ואינדקס זה הוא j, אזי לכל $k \leq i \leq j+1$ אנחנו נגדיר $B[i] = B[j]$ שכן בפרט גדולים מהכמות שם שתוארה. כך למעשה בנינו מערך B בו כל איבר מתאר את $f(i)$ שיגיד כמה איברים קטנים מi במערך. כמה עלה לנו לבנות את המערך? אנחנו חיפשנו בטבלה לכל היותר $2n$ פעמים שכן אם כל הערכים היו שונים אז חיפשנו בכל פעם ערך אחר ואז וידאנו שהוא אכן לא קיים שוב. כמו כן, ביצענו בדיוק n הוצאות. ולכן סה"כ כיוון שחיפוש והוצאה עולים $O(1)$ שילמנו $3n$, כמו כן עברנו על הטבלה B פעם אחת שעלה $O(k) = O(n)$, וכן מיינו ב$O(n)$. סה"כ עלה לנו לבנות $3n + n + n = O(n)$, כעת ניתן לבצע את השאילתא, בהינתן שני מס' נחזיר את $B[b] - B[a]$, ושוב ההנחה כמובן $1 \leq a \leq b \leq k$.

מבחן 2022 מועד ב' קיץ שאלה 3

נתון מערך A דו ממדי של מס' שלמים. מסלול נחש במערך A מוגדר להיות סדרת ערכים $a_1,...a_k$ שמקיימים את הדרישות הבאות:

לכל $1 \leq d \leq k-1$ אם הערך a_d נבחר מהמיקום $A[i,j]$ אזי a_{d+1} יכול להיות אחד הערכים שמופיעים במיקום $A[i+1,j]$ או $A[i,j+1]$ (ימינה או למטה) וגם יתקיים $|a_d - a_{d+1}| \leq 1$ (כלומר ההפרש בין כל תא יהיה עד 1, או מס' זהים או שעוקבים. אורך המסלול יהיה מס' האיברים במסלול כלומר עבור המסלול מעלה האורך הוא k.

א. יהי T פונקציה רקורסיבית שמקבלת אינדקסים (i,j) ומחזירה את אורך מסלול הנחש הארוך ביותר שמסתיים במיקום $[i,j]$ במערך A. הציגו נוסחה רקורסיבית לחישוב T.
ב. קבע פתרון תכנון דינמי הכולל סיבוכיות זמן ומקום.
פתרון: נשים לב שמסלול הנחש יכול להתחיל מכל מקום במטריצה. בכל שלב אנחנו נבחר האם לקחת את הערך או שלא. נפעיל מקסימום על שתי האפשרויות. נשים לב כי נהיה חייבים בכל שלב נתון לבדוק שהתנאים מתקיימים. לכן אנחנו נגדיר כך את הפונקציה:
א. אם $i > n$ או $j > n$ אזי נגדיר להיות מינוס אנסוף והריצה תפסק. כנ"ל על $i,j < 0$.
ב. בהינתן שאני בערך i,j . אם $|A[i-1,j]-A[i,j]| \leq 1$ נרצה שיתקיים $T[i,j] = 1+T[i-1,j]$. אם $|A[i,j-1]-A[i,j]| \leq 1$ נרצה $T[i,j] = 1+T[1,j-1]$. אם שני אי השיוונות קורים, נרצה את המקסימום מבניהם.
ג. אם לא מתקיים אי השוויון, נחזיר אחד כי המסלול יפסק.
נגדיר זאת מתמטית:

$$T[i,j] = \left\{ \begin{array}{cc} -\infty & i,j > n \vee i,j < 0 \\ max\{1+T[1,j-1], 1+T[i-1,j]\} & |A[i-1,j]-A[i,j]| \leq 1 \wedge A[i,j-1]-A[i,j]| \leq 1 \\ 1+T[i-1,j] & |A[i-1,j]-A[i,j]| \leq 1 \\ 1+T[i,j-1] & A[i,j-1]-A[i,j]| \leq 1 \\ 1 & o.w \end{array} \right\}$$

לפתרון תכנון הדינמי, נשתמש במטריצה ממימדים $n \times n$, נמלא אותה לפי האלכסונים, נשים לב כי זה יתן מילוי תא ב$O(1)$. ישנם n^2 תאים, ולכן סיבוכיות זמן ומקום $O(n^2)$. נראה כי הפתרון יכול להיות בכל אחד מתאי המטריצה, ולכן נסרוק אותה על מנת לקבל את הפתרון, נמצא בה את הערך המקסימלי ביותר. סה"כ סיימנו.

2022 מועד ב קיץ' שאלה 4

א. הוכח הפרך - משפחת עצים שמקיימת את הדרישה שלכל צומת היחס בין כמות הקודקודים בעץ השמאלי לכמות הקודקודים בעץ הימני הוא לכל היותר 2 הינה משפחה מאוזנת.
הוכחה: נוכיח באינדוקציה על מס' הקודקודים.
בסיס: אם $n = 1$ אכן הטענה מתקיימת באופן ריק שכן אין תת עץ שמאלי וימני.
צעד: נניח נכונות הטענה לכל $k < n$, כלומר לכל עץ שכזה עם k איברים מתקיים $h = O(logk)$.
יהי עץ עם n קודקודים. אזי, נסמן את תתי העצים שלו T_r וT_L. אנחנו יודעים כי העץ מקיים את הדרישה ולכן $\frac{|T_L|}{|T_R|} \leq 2$. בפרט, $|T_L| < n$ ולכן $h(T_L) = O(log|T_L|)$, באופן דומה $h(T_R) = O(log|T_R|)$. סה"כ מתקיים כי גובה העץ הינו $h = 1 + max\{h(T_R), h(T_L)\}$. כלומר

$$h = 1 + max\{h(T_R), h(T_L)\} = 1 + max\{log|T_L|, log2|T_L|\} \leq 1 + log2|T_L| \leq 1 + logn = O(logn)$$

הערה - בכל מקרה שני העצים חסומים ע"י $O(logn)$ ולכן המקסימום מבניהם. לכן סה"כ גם העץ שלנו. כנדרש.
ב. הוכח הפרך - משפחת עצים שמקיימת את הדרישה שלכל צומת כמות העלים בתת העץ השמאלי גדולה לכל היותר ב1 מכמות העלים בתת העץ הימני הינה משפחה מאוזנת.
הפרכה: נבנה שרוך שמאלי, ע"י כך שבכל שלב בירידה שמאלה נרד שני איברים מטה ויהיה לו תת עץ ימני בכל אחת מהירידות מטה שיש בו שם יהיה עלה אחד. סה"כ התנאי יתקיים. עם זאת, אם נסתכל על גובה העץ נקבל שגובהו הוא כמס' האיברים $O(n)$.
ג. יהי עץ T_1 עץ 2-3 בגובה h ויהי עץ T_2 עץ 2-3 בגובה $h+1$ אזי כמות הקודקודים בעץ T_1 בהכרח יותר קטנה מכמות הקודקודים בעץ T_2. כתוב נכון או לא נכון אין צורך בנימוק.
פתרון: לא נכון זו התחושה אך ננסה לבדוק זאת. נבנה עץ ראשון T_1 שיהיה עץ עם קודקוד ראשון בו שני ערכים ולכן מה$node$ הזה יש 3 קודקודים וסה"כ 4. הגובה הוא 1. לעומת זאת נבנה עץ שהקודקוד הראשון עם ערך אחד, כלומר יש לו שני בנים. בן ימני יהיה עלה, בן שמאלי יהיה עם בן שמאלי אחד משלו וזהו. סה"כ יש בעץ הזה 4 קודקודים גם והכמות שווה. וכן הגובה כאן הוא 2.

תל אביב - 2023 שאלה 2

א. נתון מערך באורך n עם חזרות שמכיל מספרים מהתחום $[1, 2^n]$. תארו אלגוריתם שבודק בזמן $O(n)$ בתוחלת האם קיים זוג איברים שונים $a_1 \neq a_2$ עם שכיחויות n_1, n_2 בהתאמה כך שמתקיים $\frac{a_1}{a_2} = \frac{n_2}{a_1}$
פתרון: נראה כי הנתון שקול לבדוק האם קיימים זוג איברים כך ש $a_1 n_1 = a_2 n_2$. כעת נרצה ליצור מערך שכיחויות. נעבור על המערך ונמצא את מס' האיברים השונים (למשל הכנסה לטבלת האש ותוך כדי לפני שמכניסים לחפש האם קיים האיבר לפני, אם לא אז הוא חדש ונגדיל באחד). כי נסמן ערך זה בx. נבנה מערך B בגודל x וכעת נחזיק במערך זוגות סדורים כך שהזוג יכלול (x_i, n_i) כך שx_i הוא מס' המופעים של הערך n_i. כיצד נחשב את הזוגות הסדורים של B?

נעבור על מערך A פעם אחת, ובאמצעות טבלת האש נחשב את השכיחויות ב$O(n)$ [ההסבר סזיפי אך די אינטואטיבי - נכניס לטבלה, נתחיל לחשב... יצא $O(n)$]

כעת ניגש ליצור טבלת האש בגודל x. נכניס לטבלה את הערכים x_in_i לכל $1 \leq i \leq x$, הכנסה שלהם לטבלה בגודל x תעלה $O(x)$, שזה כמובן $O(n)$ כי $x \leq n$. כעת נעבור על x האיברים שלנו בטבלה, עבור כל איבר נוציא אותו לרגע, ונבדוק האם קיים איבר בטבלה שזהה לו. אם כן, נסיים. אחרת, לא קיים. נחזיר אותו לטבלה ונמשיך להתקדם. מדוע שנוציא אותו? חיפוש יגיד לנו האם קיים, תמיד יהיה קיים כי הוא בעצמו נמצא והחיפוש בטבלת האש לא מתחשב בכמות אלא עוצר כשמוצא. לכן נוציא ואז נחזיר. הוצאה היא 1 וכן הכנסה חזרה זה 1 וכן חיפוש זה 1 ולכן 3 פעולות עבור n איברים, סה"כ $3n = O(n)$

מציאת מס' האיברים השונים $O(n)$, יצירת המערך B בזמן לינארי $O(n)$, הכנסה לטבלת האש $O(n)$, מעבר על האיברים $3n = O(n)$ וסה"כ $O(n)$.

ב. כעת נתון כי המספרים מהתחום $[n^3, n^5]$ כתוב אלגוריתם שיעבוד במקרה הגרוע.

פתרון: נרצה להשתמש בנתון בכדי למיין הפעם ולא ניפול לטריק של הנתון המיותר מסעיף א. נבחר בסיס n ואז אורך הכי גדול של מס' יהיה $d = log_n n^5 = 5$ ונקבל $O(5(n+n)) = O(10n) = O(n)$. כעת קיבלנו מערך ממויין, נרצה לעבור עליו וליצור ממנו מערך שכיחויות. נעבור עליו פעם אחת ונספור כמה איברים שונים יש (ע"י השוואה ע"י בין זוגות סמוכים), נקרא למס' הערכים השונים x. כעת ניצור מערך B כך ש$|B| = x$, כעת ניצור מערך של זוגות סדורים שבו כמו בסעיף א' כל זוג יהיה מס' מופעים וערך, כעת נוכל לחשב ערך זה ב$O(n)$, נעבור פעם אחת על המערך A כל עוד הערכים זהים נגדיל את הקאונטר שלנו, כאשר ערך כבר לא זהה נעדכן את האיבר הקודם להיות הערך עם מס' המופעים שמצאנו ונתקדם לאינדקס הבא זה עלה $O(n)$. כעת, נחשב את מערך השכיחויות. נגדיר $|C| = x$ כך שבכל תא יופיע הערך n_ix_i. נשים לב כי השכיחות הגדולה ביותר עבור איבר יכולה להיות n, וכן המס' הכי גדול יכול להיות n^5 וכן במקרה הכי גרוע האיבר מופיע פעם אחת ואז זה בדיוק $1 * n^3$ ולכן איברי C מקיימים ששייכים ל$[n^3, n^6]$ כפי שהסברנו קודם ניתן למיין ע"י בחירת בסיס n ונקבל $d = log_n n^6 = 6$ ונקבל $O(6(x+n)) = O(6x+6n) = O(n)$ כלומר מיינו את C ב$O(n)$, כעת נעבור על איברי C, אם נמצא זוג סמוך זהה אזי מצאנו זוג שמקיים שמכפלת ערכו במס' השכיחויות זהה, ונפסיק. כמובן שאם ידרשו ממש את האיבר ומס' השכיחויות ניתן לשמור את שניהם (או אפילו אחד מהם). סה"כ הזמן לינארי כחיבור של n ונקבל $O(n)$.

שאלה 3 תל אביב 2023

שאלה זו עוסקת במבנה נתונים המתחזק סדרה של ערכים חיוביים $a_1, ...a_n$ ומאפשר גישה לסכומים חלקיים מהצורה

$$S_i = \sum_{i=1}^{i} a_k$$

א. תארו מבנה נתונים התומך בפעולות הבאות -

1. $init(a_1, ...a_n)$ - אתחול המבנה ב$O(n)$
2. $SetValue(i, x)$ קביעת הערך של a_i להיות x בסיבוכיות $O(logn)$
3. $GetPartialSum(i)$- החזרת הערך של S_i בסיבוכיות $O(logn)$

פתרון:

נבנה עץ AVL. המפתחות שלו יהיו האיברים

נבנה עץ AVL. המפתחות שלו יהיו האיברים $1, ...n$ שכבר ממויינים והערכים שנשמור בו יהיו ערכי העץ. כמו כן, בכל תת עץ עד לאיבר הi יישמר שדה בשם sum. כעת נפרט את הבנייה: נבנה את העץ לפי האינדקסים $1, ..i$. בעת הבנייה, בכל שלב שבו אנחנו נכניס איבר נעדכן את הsum שלו בצורה רקורסיבית. את הראשון נגדיר $s_1 = a_1$ וכן עבור s_2 למשל נגדיר $s_2 = sum(s_1) + a_2$. וכך בצורה רקורסיבית עם הבניה, זה מעבר על n איברים פעולה של $O(1)$ ולכן לא תשנה את הסיבוכיותה אסימפטוטית. סה"כ נבנה את המבנה כך ב$O(n)$. כעת אם נרצה לקבוע את הערך של a_i להיות x, מה שנצטרך לעשות הוא לחפש את הערך a_i בעץ. נשמור אותו במשתנה זמני. כעת, נחליף אותו בערך x. נשים לב שהעץ לא השתנה ונשאר AVL כיוון שהמפתח הוא לפי i. כעת, המקרה הכי גרוע הוא כאשר a_i היה עלה. נצטרך לעדכן את כל המסלול של הערכים מהעלה עד למעלה בעץ, ונגדיל את הסכום של כל איבר כזה בx, ונחסיר את ערך האיבר הקודם שהיה (ששמרנו במשתנה עזר), עברנו לכל היותר על הגובה ולכן $O(logn)$. כעת, להחזרת הערך S_i פשוט נחפש את הערך i בעץ, מה שיעלה $O(logn)$ ונחזיר את הערך sum שלו. מש"ל.

ב. הוסיפו למבנה את הפעולה הבאה מבלי לשנות את סיבוכיות הקודמות - $SearchPartialSum$ החזרת i עבורו $S_i = x$ או $null$אם לא קיים אף i שכזה בסיבוכיות $O(logn)$.

פתרון: כעת נשתמש בעובדה שהמספרים חיוביים, לכן בהכרח הסכום הזה שאנחנו מחפשים גדל עם הזמן. הוא יהיה באמצעו מבחינת הגודל בשורש, ויהיה המקסימלי כאשר נלך לאיבר הימני ביותר. מה שנעשה יהיה להתחיל בחיפוש x בעץ. אנחנו נשווה את השורש , אם הוא קטן מאיקס נלך ימינה, אם הוא גדול מאיקס נלך שמאלה. כך נמשיך עד שנמצא אותו או שלא במידת הצורך. מה שחשוב להבין הוא שהערכים בS_i יהיו עולים שכן הערכים חיוביים ולכן הם נשמרים כאן כעץ חיפוש. סה"כ חיפוש בעץ זה יהיה שוב $O(logn)$.

ג. לכל $b \in \mathbb{R}$ נגדיר $Q_i(b) = \sum_{k=1}^{i} (a_k + b)^2$

הוסיפו למבנה הנתונים את הפעולה הבאה $GetQi(i, b)$ שתחזיר את הערך $Q_i(b)$ בזמן $O(logn)$

פתרון: נשתמש באי השוויון $(x+y)^2=x^2+y^2+2xy$. אצלנו זה אומר

$$Q_i(b)=\sum_{k=1}^{i}a_k^2+2a_kb+b^2$$

נשים לב כי b אינו קבוע ולא ידוע בזמן אתחול הנתונים לצערנו. הסכום שמאותחל לנו הוא $\sum_{k=1}^{i}a_k$. נראה כי כל שנדרש הוא לחשב את

$$\sum_{k=1}^{i}a_k^2+\sum_{k=1}^{i}2a_kb+\sum_{k=1}^{i}b^2=\sum_{k=1}^{i}a_k^2+2biS_i+ib^2$$

כלומר, מה שנעשה יהיה להוסיף למבנה שדה שיחשב את $\sum_{k=1}^{i}a_k^2$, נראה לו $sum-squre$ נניח, והוא יחושב בכניסה למבנה כמו שחושב הסכום הרגיל וזה לא ישנה את החיפוש. כעת בשביל להחזיר את הסכום אנחנו נחפש את i הנתון, יעלה לנו $O(logn)$, ואז יהיה בידינו שני הערכים $sum-squre$ וsum, ונחזיר את המשוואה הפשוטה הבאה $sum-squre+2bi*sum+ib^2$. כל זה ב$O(logn)$ כנדרש.

תל אביב 2023 מועד ב'

שאלה 1 ב'

השאלה: ומה אם באלגוריתם $select$ נחלק במקום חמישיות לשביעיות? הצג נוסחת נסיגה וחשב סיבוכיות.

פתרון - באלגוריתם סלקט המקורי חילקנו לחמישיות, ולאחר מכן בחרנו את חציון החציונים. כמה איברים הפכו ללא רלוונטים? במחצית מהאיברים, $\frac{3}{5}$ ולכן סה"כ $\frac{3n}{10}$ הפכו ללא רלוונטים. כאן אצלנו נחלק לשביעיות ואז נבחר את חציון החציונים, כמה לא יהיו רלוונטים? מחצית מהשביעייה ועוד אחד, כלומר $\frac{4}{7}n*\frac{1}{2}$ כלומר סה"כ $\frac{2n}{7}$ יהפכו ללא רלוונטים, כלומר הקלט יקטן ל$\frac{5n}{7}$. וכן תתבצע עוד עבודה של n ולכן סה"כ נוסחת הנסיגה תהיה $T(n)=T(\frac{n}{7})+T(\frac{5n}{7})+O(n)$

כעת נוכיח שסיבוכיות נוסחה זו היא $T(n)=O(n)$ באינדוקציה.

בסיס: $n=1$, נכון כמובן באופן ריק.

צעד: נניח נכונות הנוסחה לכל גודל קלט $k<n$. בפרט, קיים c קבוע כך ש$T(\frac{n}{7})=c\frac{n}{7},T(\frac{5n}{7})=c*\frac{5n}{7}$ כיוון שגודל הקלט שלהם קטן מn, ולכן נקבל כי

$$T(n)=\frac{cn}{7}+\frac{5cn}{7}+n=\frac{6cn}{7}+n\leq\frac{6cn}{7}+cn=\frac{13c}{7}n=O(n)$$

שכן המעבר באמצע התרחש כיוון שניקח $c>7$. מש"ל.

שאלה 2

נתון מערך A בגודל n עם חזרות. תאר אלגוריתם בעל זמן ריצה $O(n)$ למציאת כל האיברים שחוזרים על עצמם לפחות $\frac{n}{7}$ פעמים.

פתרון: נרצה להשתמש באלגוריתם סלקט. נפעיל את אלגוריתם סלקט על האיברים ה......$\frac{n}{7},\frac{2n}{7},\frac{3n}{7},$. סה"כ ביצענו 7 פעולות שעלו $O(n)$. כעת, נקבל שבעה מועמדים להיות האיברים שחוזרים על עצמם $\frac{n}{7}$ פעמים. נקח כל פעם אחד מהם, ונעבור באופן לינארי על המערך לבדוק מה השכיחות של אותו איבר. אם קיבלנו שגדולה שווה מ$\frac{n}{7}$ אזי נחזיר אותו ונמשיך. סה"כ 7 מעברים לינארים נוספים על המערך וסה"כ $14n=O(n)$.

סעיף ב - נתון מערך A בגודל n עם חזרות כך שהשכיחויות בו ייחודיות כלומר אין שני איברים עם אותה השכיחות. בנוסף ידוע שכל השכיחויות הם חזקה של 2. כלומר אפשרות לשכיחות היא ....$1,2,4,8$. נרצה לסדר מחדש את המערך בצורה ממויינת לפי השכיחות, כלומר תחילה יופיעו איברים נדירים יותר ובסוף שכיחים יותר. תאר אלגוריתם שרץ בזמן $O(n)$ לפתור את הבעיה

פתרון: אם לא היה מגיע הנתון של חזקות של 2 כנראה שלא היינו יכולים לפתור את הבעיה אז ננסה להעזר בזה. השכיחות הכי גדולה יכולה להיות $2^i=n$ כלומר $i=log_2n$ כלומר החזקה הכי גדולה של 2 יכולה להיות $logn$. כמו כן נשים לב כי סכום החזקות $\sum_{i=1}^{k}2^i<2^{k+1}$ ולכן, האיבר השכיח ביותר יהיה ביותר מחצי מערך. נפעיל סלקט על האיבר האמצעי בגודלו. כעת קיבלנו את האיבר השכיח ביותר. וכן כל האיברים שגדולים (והרי שווים לו) נמצאים מימין. כעת נשאר לנו למיין רק חצי מערך, נלך לבצע סלקט על חצי המערך שנותר, שוב על האיבר האמצעי בגודלו וגם הוא יכסה יותר מחצי ממה שנשאר. הפעם הסלקט יעלה לנו כגודל הקלט עליו נבצע את הסלקט, נראה כי זה יהיה $\frac{n}{2}$. כך נמשיך עד שנגיע לקלט 1 ואז המערך יהיה ממויין. כעת נראה כי נוסחת הנסיגה כאן תהיה $T(n)=T(\frac{n}{2})+O(n)$. לפי מאסטר, נקבל כי $n^{log_21}=n^0=1<O(n)$ ולכן הסיבוכיות תהיה לינארית. דרך אחרת תהיה לחשב ממש ולקבל $n+\frac{n}{2}+\frac{n}{4}+\frac{n}{8}+....=\sum_{i=1}^{logn}\frac{n}{2^i}<\sum_{n=1}^{\infty}\frac{n}{2^i}=2n=O(n)$.

שאלה 4

תאר מבנה נתונים לתחזוקת מפתחות מספריים התומך בפעולות הבאות (שימו לב דרישות הסיבוכיות הן לשיעורין ביחס לn שמציין מס' איברים במבנה)

א. $insert(x)$ הכנסת x למבנה ב$O(logn)$

ב. $Delete(x)$ מחיקת האיבר x בהינתן מצביע ב$O(1)$

ג. $Decrease(k, x)$ עבור $k < x.key$ הקטנת המפתח של x לk. נתון המצביע לx. סיבוכיות $O(1)$

ד. Min החזרת המפתח המינימלי במבנה הנתונים. $O(1)$

פתרון: נשתמש בעץ AVL. נשמור שדה בכל אחד מאיברי העץ שבו יהיה האיבר המינימלי ביותר בתת העץ שמימיניהם ומתת העץ שמשמאלהם. לממש זאת זה קל בהכנסה ובהוצאה ולא ישנה את סיבוכיות שכן נדרשים מס' החלפות על מנת לשנות. כמו כן, נוכל להכניס לעץ ערך ב$logn$ בקלות שבעץ חיפוש ולכן הכנסה היא ב$O(logn)$ ושנרצה למצוא מינימום זה יקח גם כן $O(1)$ שכן נחזיר את המינימלי מבין האיברים שמוחזקים בשורש - המינימום בתת העץ הימני והמינימום שבתת העץ השמאלי. כעת נותר לטפל בפעולה 3 - יש לנו מצביע ואנחנו משוגרים לאיבר. אנחנו רוצים להקטין את המפתח שלו. כלומר אנחנו נקטין אותו ואז נזדקק לשנות את העץ. נשים לב שהוא יתחלף רק עם כאלו שהם מתת העץ שלו השמאלי, ונראה כי לשיעורין זה יתבצע ב$O(1)$, שכן ברוב המוחלט של הפעמים אנחנו לא נעשה שינויים גדולים בגובה העץ אלא מס' החלפות קטן.

נותר לדבר על מחיקת ערך מהעץ. לכאורה מחיקה מהעץ תעלה $O(logn)$ אך האם באמת הדבר? בשביל להגיע לעץ מגובה $logn$ אנחנו צריכים לבצע קודם לפני כן n הכנסות לעץ. בפועל רוב המחיקות לממוצא לא יהיו כך. מה שנעשה יהיה לתת 2 מטבעות לכל איבר בכניסה, אחד להכנסה שלו להעץ והשני להוצאה שלו. וכך בהכנסה נשלם $2logn = O(logn)$ ובהוצאה לא נשלם כלום, סה"כ הוצאה ב$O(1)$.

2022 מועד ב' תל אביב

שאלה 1

במלון של חיים יש n חדרים. כל חדר פנוי או תפוס. עזרו לחיים לממש מבנה נתונים לתחזוקת תפוסת החדרים שיתמוך בפעולות הבאות:

א. $init(n)$ אתחול מבנה הנתונים שמתאים לn חדרים פנויים ב$O(n)$

ב. $find(i, j)$ מחזיר את אינדקס החדר הפנוי המינימלי מבין החדרים $[i, j]$ או מציין שאין.

ג. $count(i, j)$ מחזיר את מס' החדרים הפנויים בטווח החדרים $[i, j]$

ד. $set(i, b)$ אם $b = 0$ אז החדר הi תפוס ואם $b = 1$ החדר הi פנוי.

כל שאר הפעולות צריכות להיות בזמן $O(logn)$

פתרון: נרצה להשתמש בעץ AVL. נבנה אותו בזמן $O(n)$ כיוון שאנחנו למעשה רוצים לבנות עץ מהערכים $1, ...n$ ולכן נבנה רקורסיבית ע"י בחירת האמצעי בכל פעם וזה יעלה $T(n) = 2T(\frac{n}{2}) + O(1) = O(n)$, כעת המפתח של כל איבר יהיה מס' החדר. כל חדר יחזיק גם מזהה נוסף בשם b. בהתחלה לכל איבר באתחול נכניס את הb שהיה לו כאשר חיים שלח לנו את רשימת החדרים, זה כמובן לא ישנה את הסיבוכיות האסימפטוטית. נדבר על הפעולה הפשוטה set- נחפש את החדר הi בתוך העץ, ופשוט נשנה את הb שלו בהתאם ע"י הb שקיבלנו מהפעולה. זה יעלה כחיפוש בגובה העץ ולכן $O(logn)$. כעת נותר לדבר על שני הפעולות ב+ג. נרצה כעת להוסיף שדות נוספים לתוך תת העץ שלנו, נכניס את i, j אל העץ ונחשב את הדרגות שלהם בעץ $rank(i), ranl(j)$ ונחזיר את ההפרש בניהם. נזכור שאם הכנסנו אחד מהם לעץ אז להתחשב בזה בחישוב ההפרש. בסוף אנחנו נוציא אותם מהעץ. כמובן שזה יעלה $O(logn)$ שכן הדרגה היא באיזה מקום נמצא i בסדר הממויין.

כעת נדבר על הפעולה האחרונה $find$. אם i בעץ והוא פנוי נחזיר אותו. אחרת, נכניס אותו לעץ ונחפש את העוקב שלו בעץ מבין הפנוים. נסמן עוקב x. אם $x > j$ (הפנוי הבא) נחזיר שאין פנויים בטווח. אחרת, נחזיר את העוקב הפנוי x. סה"כ $O(logn)$.

2021 סמס' מועד ב שאלה 2

תכננו מבנה נתונים המכיל איברים עם מפתחות שלמים שונים ותומך בפעולות:

א. $insert(x)$ מכניס את x למבנה

ב. $delete(k)$ מוחק איבר עם מפתח k

ג. $find(k)$ בודק אם מפתח k במבנה

ד. $shift(d)$ מוסיפה את המס' השלם d לכל המפתחות במבנה

ה. $EvenSum()$ מחזירה את סכום המפתחות שערכם זוגי.

תארו מבנה שתומך בפעולות א-ג ב$O(logn)$ ובפעולות ד-ה ב$O(1)$

פתרון: נשתמש בעץ AVL. כך נקבל בחינם חיפוש הכנסה ומחיקה ב$O(logn)$. כעת נתייחס לשתי הפעולות האחרונות. נשים לב שאם נוסיף את האיבר d לכל אחד מאיברי המבנה, אזי המבנה לא ישתנה וישאר כשהיה רק עם ערכים גדולים יותר.

הרעיון יהיה כזה - נשמור משתנה עזר בשם x . בכל פעם שנבצע $shift$ נעדכן את $x+ = d$. בכל פעם שנכניס איבר למבנה, נכניס לו שדה של כמה היה השיפט כאשר הוא נכנס ואז כאשר נרצה להתייחס לערך של איבר נתייחס אליו כ $y = value + x - sumWhenInside$. כעת, פעולת $shift$ תהיה ב$O(1)$, כאשר נחפש איבר בסוף נחזיר את הערך שלו ועוד הפרש השיפטים.

כעת לפעולה האחרונה, נרצה לשמור בכל תת עץ ארבעני שדות נוספים : סכום המפתחות הזוגי מתחתיי בימין וסכום המפתחות הזוגיים מתחתיי בתת העץ השמאלי וכן סכום המפתחות האי זוגיים בתת העץ שמימיני והאי זוגיים

משמאלי. נוכל תוך כדי פעולת ההכנסה וההוצאה לעשות זאת שכן מס' השינויים שידרשו לעשות יהיה קבוע כמו שראינו בעבר בתרגילים דומים בתרגול, כעת, אם סכום הd שלי יצא זוגי אזי הוספנו d זוגי לכל אחד מהאיברים ולכן עדיין יישמרו הזוגיים ולכן נחזיר את סכום הזוגיים מימין ומשמאל של השורש. אם סכום הd יצא אי זוגי, נרצה לקחת את האי זוגיים כי אי זוגי+אי זוגי=זוגי, ולכן נחבר את האי זוגיים מימין ומשמאל עם d ונחזיר. (כמובן שנשמור גם את מס' המפתחות האלו שנדע כמה פעמים עלינו להוסיף את d)

ב. צור מבנה שמממש כל פעולות אלו בזמן $O(1)$ בתוחלת.

פתרון: כעת נוכל להשתמש בטבלת האש, טבלה לאלו עם מפתח זוגי ולמפתח אי זוגי. חיפוש הוצאה ומחיקה זה ב$O(1)$ כמובן. אם נרצה להחזיר סכום מפתחות זוגי - אנחנו נחליט שבמבנה של טבלת האש של הזוגיים אנחנו נזכור את מס' האיברים שיש לי כרגע, ואת הסכום שלהם (ניצור אותו כשנכניס אותם), וכך גם באי זוגיים. כאשר נוסיף איבר d אנחנו נשים לב אם הוא היה זוגי אזי הטבלאות היו כשהיו (זוגי+זוגי=זוגי) אי זוגי + זוגי = אי זוגי, אם אם d היה אי זוגי הטבלאות יתהפכו למעשה, ואחת תהווה של הזוגיים והשנייה של האי זוגיים. ושוב כאשר נרצה להחזיר את הזוגיים או האי זוגיים נחזיר בהתאם את הערך ששמרנו, וכאשר נרצה להוסיף d לכולם זה כמו שתיארתי מעלה. סה"כ די טריוואלי ב$O(1)$.

2021 מועד ב' תל אביב שאלה 4

נתון מערך A עם n איברים.

תארו אלגוריתם מהיר ככל היותר בתוחלת שמוצא את כל זוגות המספרים העוקבים $(k, k+1)$ שנמצאים במערך. שימו לב שהמפתחות לא בהכרח ממוקמים באינדקסים עוקבים.

פתרון: אמרו תוחלת נשתמש בטבלת האש. נעבור על האיברים במערך, ונכניס אותם לטבלת האש. בכל שלב לפני שנכניס איבר למערך נבדוק האם העוקב שלו קיים במערך או הקודם שלו. אם כן, מצאנו זוג מספרים שקיים במערך. נדפיס ונמשיך הלאה. סה"כ n הכנסות לטבלת האש ולכן $O(n)$.

ב. כעת נתעניין בתת הקבוצה בגודל המקסימלי של איברי A שמהווה אוסף איברים עוקבים. כתוב אלגוריתם מהיר ככל היותר בתוחלת שפותר את הבעיה.

פתרון: שוב אמרו תוחלת נשתמש בהאש. בנאיבי הינו ממיינים ואז עוברים על המערך והיה עולה $O(nlogn)$. ננסה לשפר -

נכניס את כל איברי המערך לתוך טבלת האש ב$O(n)$.

נשים לב שכל איבר יכול להיות ברצף יחיד ולכן עבור כל מפתח נחפש את האיברים $k+1, k+2, ...$ וכן $k-1, k-2....$. עד שנמצא איבר שלא קיים. נשמור את אורך הרצף ואם גדול מהמקסימלי שמצאנו עד כה נעדכנו. כעת נמחק את כל איברי הרצף מהטבלה שכן איבר לא יכול להיות בשני רצפים. סה"כ עברנו על כל איברי הטבלה בזמן לינארי ולכן זה $O(n)$. משל.

2017 מועד א' אלגו שאלה 5

השאלה: נתונה מטריצה בגודל $2 \times n$ של מס' טבעיים וn אבנים. ברצוננו להניח חלק מהאבנים או את כל האבנים על איברי המטריצה (כל אבן יכולה להיות מונחת על איבר אחד), הנחת אבנים היא חוקית אם אין שתי אבנים אחת מעל השנייה במטריצה או אחת ליד השנייה. כלומר אם יש אבן במיקום $[1, 1]$ לא יתכן שתהיה אבן מתחתיה ב$[2, 1]$ או מצידה כלומר $[2, 1]$. נקרא לסכום המס' הטבעיים המכוסים ע"י האבנים סכום ההנחה. מבין כל ההנחות האבנים החוקיות סכום ההנחה הגדול ביותר יקרא סכום ההנחה המקסימלי.

א. כתוב נוסחה רקורסיבית לחישוב סכום ההנחה המקסימלי

ב. הצע אלגוריתם תכנון דינמי לפתרון הבעיה על בסיס הנוסחה הרקורסיבית, נתח סיבוכיות זמן ומקום.

פתרון:

נגדיר פונקציה $f(i, j)$ כאשר i הוא מצב העמודה הj וכן $i = 0$ לא הנחנו אבן בעמודה $i = 1$ הנחנו בשורה 1 ו$i = 2$ הנחנו בשורה 2. כעת נשים לב לשלושה מצבים אפשריים:

אם $i = 0$ אזי נוכל לבחור את המקסימלי מבין לשים ב2 ב1 או לא לשים כלל. אם $i = 1$ אזי נוכל רק לדלג. נתאר זאת כך

$$
\begin{aligned}
f(0, j) = &\quad max\{f(0, j-1), f(1, j-1), f(2, j-1)\} \\
f(1, j) = &\quad A[1, j] + max\{f(0, j-1), f(2, j-1)\} \\
f(2, j) = &\quad A[2, j] + max\{f(0, j-1), f(1, j-1)\}
\end{aligned}
$$

כלומר, בכל מצב משלושת האפשרויות נפעל לפי מה שכתוב כאן. כמובן תמיד $f(0, 0) = f(1, 0) = f(2, 0) = 0$ כאשר אנחנו ננסה את כל אחת משלושת האפשרויות, כלומר ננסה לבחור שלא לשים בכלל, ננסה לבחור שלשים ב1 וננסה לבחור שלשים ב2. נשים לב שירוצו לנו כאן שלושה טבלאות בגודל $2 \times n$. לבסוף, הפתרון יהיה $max\{f(0, n), f(1, n), f(2, n)$

נבנה טבלאות ונמלא אותם משמאל לימין כפי שכתוב בנוסחאות, לבסוף כפי שתיארתי הפתרון יהיה $max\{f(0, n), f(1, n), f(2, n)$, סה"כ סיבוכיות הזמן תהיה לינארית שכן נעבור על $3*2*n = 6n = O(n)$ תאים וכך גם סיבוכיות המקום תהיה לינארית. כנדרש.

אלגו' 1 מועד א' קיץ 2017 שאלה 4

תארו אלגוריתם מהיר אשר בהינתן $n \in \mathbb{N}$ מחשב את $f(n)$ המוגדרת:

$$f(n)=\left\{\begin{array}{cc} 1 & n\in\{0,1,2\} \\ f(n-2)+2f(n-3) & o.w \end{array}\right\}$$

פתרון: מטרת הפתרון תהיה לחשב באמצעות מטריצות ב$O(logn)$ ע"י העלאה בחזקה. כעת נראה כי

$$f(n)=0f(n-1)+1f(n-2)+2f(n-3)$$

$$f(n-1)=1f(n-1)+0f(n-2)+0f(n-3)$$

$$f(n-2)=0f(n-1)+1f(n-2)+0f(n-3)$$

כלומר,

$$\begin{pmatrix} f(n) \\ f(n-1) \\ f(n-2) \end{pmatrix}=\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}*\begin{pmatrix} f(n-1) \\ f(n-2) \\ f(n-3) \end{pmatrix}$$

כעת נראה כי עבור חזקה k מתקיים

$$\begin{pmatrix} f(n) \\ f(n-1) \\ f(n-2) \end{pmatrix}=\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^k*\begin{pmatrix} f(n-k-1) \\ f(n-k-2) \\ f(n-k-3) \end{pmatrix}$$

צ"ל $n-k-3\geq 0$ ולכן $k\geq n-3$. בפרט יתקיים

$$\begin{pmatrix} f(n) \\ f(n-1) \\ f(n-2) \end{pmatrix}=\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{n-3}*\begin{pmatrix} f(2) \\ f(1) \\ f(0) \end{pmatrix}$$

ולכן

$$\begin{pmatrix} f(n) \\ f(n-1) \\ f(n-2) \end{pmatrix}=\begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{n-3}*\begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

העלאה בחזקה עולה $O(logn)$ והכפלה בוקטור $O(1)$ ולכן מצאנו את $f(n)$ ב$O(logn)$. מש"ל.

שאלה 5 אלגו' 1 מועד קיץ

תכננו אלגוריתם תכנות דינמי כך שבהינתן קבוצה של מס' שלמים $S=\{S_0=0\leq s_1\leq s_2\leq ...\leq s_n=m\}$ יקבע האם ניתן למקם תחנות דלק על כביש מהיר באורך m כך ש

א. ניתן למקם תחנת דלק רק במרחק $s_i\in S$ מתחילת הכביש המהיר (כלומר רק מיקומים מהקבוצה)

ב. חייבת להיות תחנת דלק בתחילת הכביש בתחנה s_0 ובסופו s_n

ג. המרחק בין שתי תחנות עוקבות הוא בין 15 ל25 .

למשל ־ אם $S=\{0,15,40,50,60\}$ האלגוריתם יחזיר כן כי $\{0,15,40,60\}$ הוא פתרון אפשרי.

פתרון:

ננסה לחשוב כי מרגיש מאתגר. תמיד האיבר הראשון 0 וm יהיו בהם תחנות. כעת המטרה תהיה להגדיר פונקציה f כך שבהינתן שאני בתחנה מסוימת i $f(i)$ תחזיר לי האם ניתן למקם תחנות עד לתחנה הנוכחית. בעצם מה שנעשה בכל שלב נתון יהיה לבדוק האם קיימת תחנה j כך ש$S[i]>S[j]$ וכן יתקיים שהמרחק ממנה הוא בטווח. כיצד נדע שהיא בתוך הבחירה? נשתמש בf שלה. נתאר זאת כך כדקלמן:

$$f(i) = \quad \vee_{1\leq j\leq i}\{f(j)\} \quad 15 \leq S[i]-S[j] \leq 25$$

כלומר נבדוק בכל שלב, האם קיימת כזו כך שהפרש המרחקים יהיה בתחום הנדרש. אם נמצא כזו, נבדוק אם היא בפתרון עד כה עם $f(j)$. יחזיר $true$? נפסיק לחפש כזו ונתקדם הלאה. אם $i=n$ אנחנו נבדוק האם קיימת תחנה שהמרחק שלה ממני הוא בטווח, וכן בוודאות התחנה האחרונה תהיה בפתרון הסופי.
הפתרון לתכנון הדינמי יהיה ליצור מערך בגודל n. בכל שלב אנחנו נמלא אותה לפי נוסחת הנסיגה המפורטת מעלה. מילוי כל תא יעלה במקרה הגרוע $O(n)$ ולכן מילוי n תאים יעלה $O(n^2)$, לכן סה"כ סיבוכיות הזמן ריבועית והמקום לינארית $O(n)$. הפתרון יהיה בתא $A[n]$ בו יאוחסן $f(n)$. ניתן להוריד את סיבוכיות המקום ל$O(1)$ אם בכל פעם "נזרוק" את כל התחנות שמרחקן גדול מ25 ואינן רלוונטיות עוד.

שאלה 5 2016 אלגו מועד א

השאלה: סדרת מספרים $R[1,...,m]$ תקרא **דו עולה** אם לכל $i>2$ מתקיים $R[i]>R[i-2]$. למשל $5,2,7,3,11,9,15,14$. במילים אחרות, סדרה דו עולה מכילה בצורה מתחלפת שתי סדרות עולות.
הפלט נדרש להיות אורך תת סדרה דו עולה ארוכה ביותר של A.
הצע נוסחת נסיגה והשתמש בה לפתרון תכנון דינמי. נתח סיבוכיות זמן ומקום.

פתרון:

נרצה להגדיר פונקציה f שתחזיר את אורך תת הסדרה הארוכה ביותר כאשר אנחנו מסתכלים על המספרים $1,...i$ בתוך המערך R. לכן נגדיר $f(i)$ כפונקציה שתחזיר את אורך תת הסדרה הדו עולה במערך R עבור המספרים $1,...i$ **שכוללת את המס' i**. ואז הפתרון לבעיה יהיה לסרוק את כל $f(1),...f(n)$ ולחפש מקסימום מתוכם. נשים לב לתנאי בסיס. אם $i=1$ אזי תמיד $f(1)=1$ שכן איבר בודד הוא אכן תת סדרה עולה. כמו כן נשים לב כי אם $i=2$ גם כן נגדיר $f(2)=2$ כי סדרה באורך 2 היא אכן לפי ההגדרה באופן ריק סדרה דו עולה. כעת לכל $i>2$ נבדוק לפי תנאי הבסיס ונגדיר את הנוסחה הבאה:

$$f(i)=\left\{\begin{array}{cc} 1 & i=1 \\ 2 & i=2 \\ f(i-1)+1 & i>2 \wedge R[i]>R[i-2] \wedge f(i-1)>1 \\ 2 & i>2 \wedge R[i]\neq R[i-1] \\ 1 & o.w \end{array}\right\}$$

הסבר: אם $i=1$ או $i=2$ הסברתי מעלה. אם אכן האורך גדול מאחד וכן מתקיים העלייה המדוברת אזי נרצה להחזיר את האורך הקודם ועוד אחד. כעת נשים לב שסדרה שכזו היא באורך 2 לפחות, פרט למצב אחד בו האיברים זהים זה לזה ברצף לכן במקרה כזה תמיד אם לא מתקיים $R[i]>R[i-2]$ אזי בהכרח אורך תת הסדרה הארוכה ביותר שמכילה את האיבר הi תהיה באורך 2 . סה"כ הנוסחה ברורה.
כעת ניצור מערך באורך n ונקרא לו A. נמלא אותו לפי הנוסחה המתוארת. סה"כ נמלא כל תא ב$O(1)$ וכן יש n תאים ולכן נבצע $O(n)$ עבודה. לבסוף, נרצה לסרוק את המערך על מנת למצוא את תת הסדרה המקסימלית וזה יעלה עוד $O(n)$ עבודה. סה"כ סיבוכיות הזמן והמקום הינה לינארית וכן אי אפשר להקטין לצערנו את השימוש במקום, שכן לבסוף צריך לסרוק את המערך.

שאלה 6 2016 קורס קיץ אלגו 1

השאלה: נתונה מטריצה M בגודל $n\times n$ המאחסנת בכל תא מס' חיובי שיהווה את מחיר התא. מסלול נסיעה מתא כלשהו מתחיל בעמודה השמאלית, עד לתא כלשהו בעמודה הימנית. המסלול יקרא חוקי אם יתקיים:
א. המסלול יכיל תא אחד בדיוק באותה עמודה. כלומר, אי אפשר לעלות למעלה ולמטה.
ב. המעבר מעמודה k לעמודה $k+1$ יכול להיות ימינה, ימינה מעלה או ימינה למטה.
נרצה את המחיר המינימלי למסלול חוקי כלשהו מתא בעמודה השמאלית לתא בעמודה הימנית. כתוב נוסחה רקורסיבית ופתרון תכנון דינמי. חשב סיבוכיות זמן ומקום.
פתרון: נשים לב שניתן להתחיל מכל תא על העמודה השמאלית ביותר. נגדיר $f(i,j)$ כפונקציה שתחזיר את המחיר המינימלי למסלול עד לתא $f(i,j)$ ונשים לב למקרי הקצה הבאים:
א. אם אנחנו בתא כלשהו במטריצה שלא עושה בעיות, ניתן ללכת לתאים: $f(i,j+1),f(i+1,j+1),f(i-1,j+1)$ ולבחור את המינימלי מביניהם.
ב. אם $j=n$ אזי שסיימנו את המסלול והגענו לעמודה הימנית ביותר. אם התקדמנו נגדיר אנסוף.
ג. אם $i=1$ אזי אנחנו בשורה הראשונה, ניתן רק ללכת ימינה או למטה $f(i,j+1),f(i+1,j+1)$
ד. אם $i=n$ אזי אנחנו בשורה האחרונה, ניתן ללכת רק ימינה או מעלה $f(i,j+1),f(i-1,j+1)$
נשים לב שאנחנו ממלאים את המטריצה מההתחלה לסוף לכן נתאים את התנאים שנכתבו כאן לפתרון רקורסיבי שכזה. לכן נגדיר נגדיר את הנוסחה הבאה:

$$f(i,j)=\left\{\begin{array}{cc} \infty & j>n \\ min\{(i,j-1)+A[i,j],f(i-1,j-1)+A[i,j]\} & i=1 \\ min\{f(i,j-1)+A[i,j],f(i+1,j-1)+A[i,j]\} & i=n \\ min\{f(i,j-1)+A[i,j],f(i-1,j-1)+A[i,j],f(i+1,j-1)+A[i,j]\} & o.w \\ A[i,1] & j=1 \end{array}\right\}$$

נשים לב שכל אחד מהאפשרויות בעמודה הראשונה יכול להיות פתרון. לכן נגדיר תנאי בסיס אם $j=1$ הערך יהיה $A[i,1]$

סה"כ נבנה מטריצה $n\times n$. נתחיל במילוי העמודה הראשונה, ולאחר מכן נשים לב שבהינתן שאני בתא מסויים, אני זקוק לעמודה הקודמת שלי ולכן נמלא את המטריצה לפי עמודות. סה"כ מילוי כל תא יעלה $O(1)$ שכן נמלא מלמעלה למטה את המטריצה בעמודות. יש n^2 תאים ולכן סיבוכיות הפתרון תהיה $O(n^2)$ זמן. גם מקום יהיה ריבועי. לכאורה לא מצאנו את הפתרון. מה שנצטרך יהיה לעבור על עמודות המטריצה כאשר $j=n$ ולמצוא מהן את המינימלי. זה יעלה $O(n)$ לסריקה. סה"כ נוכל גם לצמצם שימוש בזכרון ל$O(n)$ אם בכל שלב נשמור רק שתי עמודות אחרונות (נוכחית וקודמת) שכן אין צורך ביותר מזה. לכן סה"כ סייבוכיות הזמן היא $O(n^2)$ והמקום $O(n)$.

אלגו 1 2018 מועד ב שאלה 5

השאלה: סדרה של מספרים $X=x_1,...x_n$ תקרא סדרת זיג זג אם מתקיים $x_1<x_2>x_3<x_4.....$. בהינתן מערך $A=a_1,....a_n$ תאר אלגוריתם תכנון דינמי שמוצא את אורך תת הסדרה הארוכה ביותר של A שהינה סדרת זיגזג.

פתרון:

הפתרון הנאיבי יהיה לעבור על 2^n תתי הסדרות ולבדוק כל אחת מהן ונגיע לפתרון שהינו $O(n*2^n)$. לא בא בחשבון.

נרצה להשתמש בקריאות חוזרות ולכן נגדיר פונקציה f ונגיע לתובנה ראשונה - כל איבר בודד בפני עצמו הוא סדרת זיג־זג. נגדיר את $f(i,s)$ להיות הפונקציה שמחזירה את תת הסדרה הארוכה ביותר שהינה סדרת זיג־זג וכן a_i חלק ממנה וכן s הוא המצב האחרון שהיינו בו כאשר $s\in\{0,1\}$ ש0 מציין ירידה ו1 מציין שכעת אנחנו צריכים לעלות. מכאן נגדיר את נוסחת הנסיגה הבאה:

$$f(i,s):=max_{1\leq k<i}\left\{\begin{array}{c} f(k,0)+1 \\ f(k,1)+1 \end{array}\right\}\begin{array}{cc} state=1 & A[k]<A[i] \\ state=0 & A[k]>A[i] \end{array}$$

ותנאי נוסף לנוסחה יהיה אחרת, תגדיר פשוט אחד. מדוע הגדרה זו עונה על הנדרש? כיוון שאם אני כרגע במצב $state=1$, אני צריך לחפש מישהו לפניי שקטן ממני וכן הסטייט שלו הינו 0 שכן אנחנו רוצים לזגזג. באופן דומה על $state=0$.

כעת ניצור מערך דו ממדי בגודל $n\times 2$ ונקרא לו B. נתחיל למלא אותו משמאל לימין לפי סדר המילוי מנוסחת הנסיגה כאשר נגדיר תנאי בסיס, $f(0,0)=f(0,1)=1$ שכן כל איבר בודד הוא בפרט סדרת זיג זג. נמלא באמצעות שימוש בתאים קודמים. נראה כי מהנוסחה ניתן לראות שמילוי כל תא יקח במקרה הגרוע ביותר $O(n)$ ולכן סיבוכיות זמן הריצה שלנו תהיה $O(n)$ כפול מס' התאים ונגיע ל$O(n^2)$ וכן סיבוכיות המקום תהיה $O(n)$. כמו כן, נראה כי בסוף נאלץ לסרוק את המערך למצוא את אורך תת הסדרה הארוכה ביותר כיוון שההגדרה שלנו היא שהאיבר הi משתתף, והסריקה תעלה עוד $O(n)$ וסה"כ נמצא את תת הסדרה הארוכה ביותר שהינה זיגזג. מש"ל.

אלגו 1 2017 תרגיל 6 שאלה 1

הרשלה מעוניין להשאיל ספרים מספריית האוניברסיטה אך בספריה יש מגבלת השאלה של עד k ספרים לאדם. לאור זאת ברגע שהרשלה מעוניין להשאיל את הספר ה$k+1$ הוא נאלץ להחזיר ספר אחר. באופן זה יכול להיווצר מצב שעל הרשלה להשאיל שוב ספר שהיה ברשותו והוחזר.

נתונה סדרה של n ספרים $a_1,....a_n$אותם הרשלה מעוניין להשאיל בזה אחר זה. הסדרה מסודרת כבר לפי סדר הזמנים בו הרשלה מעוניין להשאילם כאשר יתכן וחלק מהספרים בסדרה חוזרים על עצמם. כתוב אלגוריתם חמדני הממזער את כמות ההחזרות של ספרים שהרשלה יבצע אם מותר להשאיל בו זמנית לכל היותר k ספרים.

פתרון: האלגוריתם יפעול כך: אם מס' הספרים קטן מk, נמשיך לקחת ספרים. אחרת, במידה ואנחנו חייבים להחזיר ספר, נבדוק מהספרים הנוכחיים איזה מהספרים יופיע שוב הכי מאוחר ונחזיר אותו. ובלי קשר - אם נמצא ספר שכלל לא יופיע שוב בהמשך וודאי שנחזיר אותו שכן הוא יופיע הכי מאוחר (הוא לא יופיע כלל......). ניגש להוכחת שני הלמות הבאות:

א. למת הבחירה החמדנית: קיים פתרון אופטימלי לבעיית הספרים כאשר בכל פעם נחזיר את הספר שיופיע שוב הכי מאוחר.

הוכחה: אם במצב הנוכחי יש פחות מk ספרים אזי לא נחזיר כלום שכן אין צורך לכן ההוכחה תדון במצב בו מס' הספרים הנוכחי גדול מk. יהי k ספרים אצל הרשלה. כעת נסמן בb_i את הספר שמועד הביקוש שלו מחדש הכי רחוק. נסתכל על הפתרון האופטימלי OPT. אם $b_i\in OPT$ אזי סיימנו שכן אכן הספר נמצא בפתרון האופטימלי. אחרת, קיים איזשהו ספר b_j כך שמועד הביקוש שלו מחדש קצר יותר ממועד הביקוש של ספר b_i כלומר $|b_j-time|<|b_i-time|$. כעת נסתכל על הפתרון הבא: $OPT'=OPT/\{b_j\}\cup\{b_i\}$. נרצה להוכיח שהפתרון הינו חוקי ואופטימלי.

חוקי: אכן אם היו כעת k ספרים בדיוק, הוצאנו את הספר b_j וקיבלנו שבפתרון האופטימלי יש $|opt|-1$ ספרים וכן הוספנו את b_i וקיבלנו שיש כעת בדיוק את מס' הספרים התקין כלומר ביד של הרשלה אין יותר מk ספרים כי הוצאנו ספר אחד מהפתרון אך במקומו הכנסנו אחד אחר.
אופטימלי ־ בהכרח, זמן הביקוש מחדש של b_j היה לפני זמן הביקוש של b_i כלומר $|b_j - time| < |b_i - time|$, ולכן מס' ההחזרות ב$OPT$ כעת יקטן שכן יותר זמן נוכל שלא לבצע החלפות.
ב. למת תת המבנה האופטימלי: פתרון שמורכב מבחירה של הספר שמועד בקשתו מחדש הוא המאוחר ביותר בתוספת פתרון אופטימלי לבעיות הספרים עם כל הספרים לאחר הספר הזה הוא פתרון אופטימלי לבעיה.
הוכחה: נב"ש שקיים פתרון אופטימלי טוב יותר, B. כך שמס' ההחלפות בB קטן ממס' ההחלפות בA כלומר $|A| > |B|$. לפי ההגדרה בהכרח $b_i \in A, b_i \in B$ כי שניהם פתרונות לבעיה. נסתכל על $A/\{b_i\}$ ו$B/\{b_i\}$. בהכרח מההנחה $A/\{b_i\}$ הוא פתרון אופטימלי לבעיה, ולכן $|A/\{b_i\}| \leq |B/\{b_i\}|$ כלומר $|A| - |\{b_i\}| \leq |B| - |\{b_i\}|$ כלומר $A \leq B$ בסתירה ל$A > B$.
סה"כ שילוב הטענות מוביל למסקנה שנקבל פתרון אופטימלי. כעת, ניגש לדבר על סיבוכיות הפתרון: אנחנו נצטרך בכל שלב לבדוק את הספר האחרון בA שיוחזר ולחפש אותו במערך של k הנוכחיים, מה שיעלה $O(k)$ כפול n ספרים במקרה הגרוע ולכן $O(nk)$.

מבני נתונים 2010 מועד ב' שאלה 5

השאלה: בתרגול למדנו מערך דינמי שגדל כל פעם פי שניים והתבוננו בפעולת הכנסה בלבד. כעת נתייחס למערך מלא בגודל n ונניח שבמצב ההתחלתי הוא מלא לגמרי. נתבונן בפעולת הוצאה בלבד (הוצאת איבר אחרון במערך) כאשר במערך נותרו רק $\frac{n}{2}$ איברים בלבד נקצה מערך חדש בגודל $\frac{n}{2}$ ונעתיק את האיברים למערך החדש ונשחרר את המערך הישן. השתמש בפונקציה הבאה $\phi(D_i) = size - num$ כאשר $size$ הוא גודל נוכחי של מערך וnum הוא מס' האיברים כרגע במערך

א. הוכח לכל i $\phi(D_i) > \phi(D_0)$ והנח שלשם כך כי במצב ההתחלתי $D_0 = n$ והוא מלא. הסק מכך כי לכל m מתקיים $\sum_{i=1}^{m} \hat{c_i} \geq \sum_{i=1}^{m} c_i$.

ב. הנח סדרה של m פעולות הוצאה. מה העלות לשיעורין של כל פעולה? הוכח.

פתרון:

במצב ההתחלתי מתקיים כי $size = num = n$ ולכן $\phi(D_0) = n - n = 0$. כעת נוכיח כי לכל $i > 0$ מתקיים $\phi(D_i) \geq 0$. ובכן, שקול להוכיח כי בכל רגע i $size - num \geq 0$ כלומר $size \geq num$. ובכן, זה טריוואלי כיוון שתמיד מס' האיברים שיאוחסנו במערך, יהיה קטן או שווה מגודל המערך. כנדרש. כעת,

$$\sum_{i=1}^{m} \hat{c_i} = \sum_{i=1}^{m} (c_i + \phi(D_i) - \phi(D_{i-1})) = \sum_{i=1}^{m} c_i + \phi(D_n) - \phi(D_0) \geq \sum_{i=1}^{m} c_i$$

כאשר המעבר $\geq$ התבצע בשל מה שהוכחנו.

ב. נניח סדרה של m פעולות הוצאה מהמערך ונחלק לשני מקרים:

1. הפעולה הi לא גורמת להקטנת מבנה הנתונים. אזי, במצב כזה $size_i = size_{i-1}$ וכן $num_i = num_{i-1} - 1$. כלומר

$$\phi(D_i) - \phi(D_{i-1}) = size_i - num_i - (size_{i-1} - num_{i-1}) = num_{i-1} - num_i = 1$$

וכן $c_i = 1$ ולכן

$$\hat{c_i} \leq 2$$

2. הפעולה הi כן גורמת להקטנת מבנה הנתונים: כלומר הורדנו את מס' האיברים וכעת הגענו ל$\frac{n}{2}$ וצריך לצמצם. כעת $size_i = \frac{size_{i-1}}{2}$ וכן $num_i = num_{i-1} - 1$

$$\phi(D_i) - \phi(D_{i-1}) = size_i - num_i - (size_{i-1} - num_{i-1}) = 1 + \frac{size_{i-1}}{2} - size_{i-1} = 1 - \frac{size_{i-1}}{2}$$

וכן $c_i = \frac{size_{i-1}}{2}$ שכן זה העלות להעתקת האיברים, ולכן

$$\hat{c_i} = c_i \, \phi(D_i) - \phi(D_{i-1}) = 1$$

סה"כ עבור סדרה של m פעולות, לכל פעולה מתקיים $c_i \leq 2$ ולכן $\frac{T(m)}{m} \leq 2$.

מבחן שמואל קליין 2011 שאלה 1

עץ פיבונאצ'י מסדר n יסומן T_n ויוגדר ע"י T_0 העץ הריק, T_1 עץ שמכיל קודקוד אחד, ולכל $n > 1$ יתקיים T_nהוא עץ בינארי עם שורש ובן שמאלי שיהיה T_{n-2} ובן ימני שהינו T_{n-1}.

א. הוכח באינדוקציה על מבנה העץ כי מס' הקודקודים בעץ T_n הוא $F_{n+2} - 1$ כאשר F_i היא סדרת פיבונאצ'י המוכרת המוגדרת ע"י $F_0 = 0, F_1 = 1, F_i = F_{i-1} + F_{i-2}$.

ב. מה הקשר בין עצי פיבונאצ'י לעצי AVL?

פתרון:

א. נוכיח באינדוקציה על מבנה העץ את הטענה.

בסיס: $n = 0$ אזי אכן בעץ בודד מההגדרה מס' הקודקודים הוא אפס, ואכן $F_{0+2} = F_2 = F_1 + F_0 = 1 + 0 = 1$ ואכן $F_{n+2} - 1 = 1 - 1 = 0$ שזה מס' הקודקודים בעץ.

נוכיח גם עבור $n = 1$ ויש קודקוד יחיד וכן $F_{2+1} - 1 = F_3 - 1 = F_1 + F_2 - 1 = 1 + 1 - 1 = 1$ כנדרש.

צעד: נניח שהטענה נכונה לכל עץ עם מס' קודקודים קטן מn. נוכיח את הטענה על n. יהי עץ פיבונאצ'י T_n מסדר n. אזי, קיימים לו שני בנים (נניח שאינו העץ הריק והעץ הבודד שכן אלו מקרי הבסיס) שהינם T_{n-1}מימין וT_{n-2} משמאל. שני העצים האלו מקיימים את הנחת האינדוקציה. כמו כן, $|T_n| = |T_{n-1}| + |T_{n-2}| + 1$. כעת, אנחנו יודעים ש$|T_{n-1}| = F_{n-1+2} - 1 = F_{n+1} - 1$ מהנחת האינדוקציה וכן ש$|T_{n-2}| = F_{n-2+2} - 1 = F_n - 1$. סה"כ נקבל כי

$$|T_n| = |T_{n-1}| + |T_{n-2}| + 1 = F_n - 1 + F_{n+1} - 1 + 1 = F_n + F_{n+1} - 1 = F_{n+2} - 1$$

כמו כן המעבר מימין $F_n + F_{n+1} = F_{n+2}$ נבע מהגדרת סדרת פיבונאצ'י. כנדרש.

ב. כאשר הוכחנו בהרצאה שעצי AVL הם מאוזנים הגדרנו נוסחת נסיגה שחישבה את מס' הקודקודים המינימלי בעץ AVL מגובה h וקיבלנו $f(0) = 0, f(1) = 1$ וכן

$$f(h) = 1 + f(h-1) + f(h-2)$$

הנוסחה של מס' הקודקודים מאוד זהה לנוסחה אצלנו. מס' הקודקודים בשני העצים יהיה זהה (כלומר עץ פיבונאצי הוא עץ AVL מגובה n עם מס' קודקודים מינימלי) , כלומר מס' קודקודים של AVL מינימלי בגובה h הוא $F_{n+3} - 1$ שכן הפער נובע מכך שפיבונאצי מוגדר בהתחלה כעץ ריק ואצלנו בavl כעץ עם קודקוד בודד.

מבחן שמואל קליין 2011 מועד א שאלה 4

ברצוננו להכניס 667 איברים לטבלת האש בגודל M עם יוניפורם האשינג.

א. מה צריך להיות גודל הטבלה בשביל להגיע ל1.333 השוואות בממוצע עבור הכנסת האיבר האחרון? **1.** 1334 **2.** 2667 **3.** 3000 **4.** 3333.

ב. אם נבחר $M = 1000$ מה יהיה הזמן הממוצע לחיפוש מוצלח כאשר לכל איבר שהוכנס הסתברות שווה להופיע בחיפוש?

פתרון:

נתחיל מסעיף ב' כי הוא קל יותר, אנחנו יודעים כי הזמן יהיה $\frac{1}{a}ln(\frac{1}{1-a})$ וכן $\frac{2}{3} \approx \frac{667}{1000} = a$ ולכן $= \frac{1}{\frac{2}{3}}ln(\frac{1}{1-\frac{2}{3}}) = \frac{3}{2}ln(\frac{1}{\frac{1}{3}}) = \frac{3}{2}ln(3)$.

נעבור לסעיף א'. כעת נרצה להכניס איבר שעוד לא קיים כלומר $\frac{1}{1-a} = \frac{4}{3}$ נשווה ונקבל $a = \frac{1}{4}$ כלומר $\frac{667}{M} = \frac{1}{4}$ ולכן $M = 667 * 4 = 2668$ ולכן התשובה היא 2667.

מבחן שמואל קליין 2011 מועד א שאלה 5

השאלה: נתון מערך A של n מספרים לא ממויינים. עלינו לבצע עיבוד מקדים בזמן לינארי כך שלאחריו נוכל לענות על שאילתה (L, R) כך ש $1 \leq L < R \leq n$ ונוכל להחזיר את הסכום של תתא המערך $A[L, ...R]$ בזמן $O(1)$.

פתרון: נגדיר מערך עזר B שלכל תא i ישמור את הסכום $\sum_{k=1}^{i} A[k]$. כלומר, כל תא ישמור את סכום כל האיברים עד לאיבר הנוכחי. נראה כי נוכל למלא את המערך הזה בקלות לפי הנוסחה הבאה: $B[1] = A[1]$ וכן לכל $i > 1$ יתקיים $B[i] = B[i-1] + A[i]$. סה"כ נמלא את המערך במעבר לינארי על איבריו. כעת, נוכל להחזיר את השאילתא ע"י החזרת $B[R] - B[L-1]$ ב$O(1)$.

2021 סמסטר ב' מועד א' תל אביב

שאלה 4:

נרצה להוסיף ל$Union - find$ פעולה $print$ כך שבהינתן x תדפיס את כל איברי הקבוצה. תאר מבנה זה והשינויים בו

פתרון: השינוי שנעשה יהיה ליצור רשימה מקושרת לכל קבוצה. כאשר נעשה $makeSet$ ניצור גם רשימה מקושרת בגודל 1 וכן כאשר נאחד קבוצות ב$Union$ נאחד את הרשימות ע"י שינוי מצביעים. כעת כאשר נרצה להדפיס את כל איברי הקבוצה נחפש את x עם $find$ ונדפיס את כל איברי הרשימה המקושרת שלו בזמן לינארי $O(n)$. סה"כ חוץ מזה אין שינויים והמבנה גם לא השתנה אסימפטוטית. מה כן? המקום גדל, אך אסימפטוטית שוב זה לא שינה את המקום.

שאלה 6:

מימשו מבנה נתונים S שיכיל מפתחות שונים זה מזה ויתמוך בפעולות הבאות.

א. $insert(k)$־מכניס מפתח k למבנה

ב. $delete(k)$ מוחק מפתח k ממבנה

ג. $find(k)$ מחזיר $true$ אם קיים במבנה ושקר אחרת.

ד. $DistSum(k)$־ עבור מפתח k שנמצא במבנה, מחזירה את סכום המרחקים מk של מפתחות במבנה שקטנים מk. כלומר

$$\sum_{k' \in S : k' < k} k - k'$$

ב$O(logn)$ הכל כמובן.

פתרון:

הפעולות א־ג טריוואליות לכל עץ AVL שנקח ולכן נשתמש בעץ AVL ונקבל בחינם את א ב וג'. כעת לפעולה האחרונה ־ אם נפרק זאת נראה שמה שצריך להחזיר, בהינתן שx זה מס' המפתחות שקטנים מk, עלינו להחזיר את $xk - \sum k'$. כלומר, אם נשמור בכל $node$ בעץ שני שדות: שדה של כמה איברים יש מתחתיי, וסכום האיברים מתחתיי הרי שסיימנו. מדוע? עץ הAVL שנשתמש בו יהיה עץ חיפוש.כעת, נשמור לכל איבר את שדה $size$ שיקבע כמה איברים יש בתת העץ, וכן שדה sum. כעת, נוכל לתחזק את שני השדות ע"י שינוי מס' קבוע של פעולות וזה לא יפגע בסיבוכיות האסימפטוטית $O(logn)$ של הכנסה והוצאה. הפעולה $size$ תקבע כמה איברים יש בתת העץ שמושרש באיבר מסויים כנ"ל על sum. נראה כעת כי מתקיים

$$\sum_{k' \in S : k' < k} k - k' = k \sum 1 - \sum k' = kRank(k) - sum(k')$$

מי זה $sum(k')$? סכום כל המפתחות שקטנים מk, וכן אנחנו יודעים זאת ולכן נחזיר אותו וכן אנחנו גם יודעים את מס' האיברים $Rank$ וסה"כ הפתרון יהיה ב$O(logn)$

הערה: חישוב כל האיברים שקטנים מk מסויים יתבצע באמצעות השדה sum, כאשר בכל שלב נרד ונקח את הסכום. כך למשל אם איבר שלי הוא $k = 17$ ושורש העץ הוא 8, נקח את $rank(Root)$ ונלך ימינה רקורסיבית וכנ"ל על sum.

מבחן 2024 גלעד וטלייה מועד א'

שאלה 1

א. בנה מבנה נתונים שתומך בפעולות הבאות:

א. $init$ ב$O(1)$

ב. $insert(x)$ בזמן $O(logn)$

ג. $ExtractMin$ החזרת ערך המינימום במבנה ב$O(1)$

ד. החזרת ערך המינימום במבנה ומחיקתו בעלות $O(logn)$

ה. $Delete(x)$ מחיקת האיבר x מהמבנה אם הוא קיים.

פתרון:

נשתמש בעץ AVL ונשמור בו שדה min שישמור את האיבר המינימלי ביותר בתת העץ המושרש. נעיר מראש שהשימוש חוקי ולגיטימי כיוון שפעולות מחיקה/הכנסה לא ישנו את הסיבוכיות האסימפטוטית גם אם נוסיף את השדה שכן מס' השינויים יהיה קבוע.

א. $init$ ־ כיוון שאין איברים נאתחל עץ avl ריק וזה אכן יעלה $O(1)$

ב. נכניס לעץ avl כרגיל, במידת הצורך אם הכנסנו איבר שקטן מהמינימום נשנה את השדות וכפי שהערתי מדובר במס' פעולות קבוע ולכן הסיבוכיות תהיה $O(logn + c) = O(logn)$ כאשר $logn$ זה הכנסה לעץ וc זה מס' הפעולות הקבוע שנשנה.

ג. החזרת ערך המינימום תהיה : בהינתן שלשורש ששמרנו יש $minR$ ו$minL$ שהם המינימום בתת עץ מימין ותת עץ משמאל, קודם כל כיוון שזה עץ חיפוש בהכרח צד שמאל קטן יותר ולכן נוכל להחזיר את $minL$ אך בכל מקרה שקול ונחזיר את $min\{minL, minR\}$
ד. מדובר בעץ avl וכן אנחנו יודעים את איבר המינימום (מסעיף ג') ולכן נחפש אותו ראשית ב$O(logn)$ ואז נמחק אותו מהעץ ב$O(logn)$ כמו שמוחקים מעץ avl רגיל. שוב נעיר שמס' השינויים שנצטרך לעשות על מנת לשנות את השדה שהוספנו יהיה קבוע כפי שכבר ראינו בתרגול. סה"כ $2logn + c = O(logn)$
ה. כעת נמחק מהעץ כפי שמוחקים בעץ AVL רגיל , שקול לגמרי לסעיף ד' רק שהפעם במקום לחפש את min נחפש את x ואז נמחוק. סה"כ גם כאן $O(logn)$. מש"ל.

סעיף ב' =10 נקודות

תארו אלגוריתם שבהינתן מערך A ומס' k ידפיס את כל הערכים המינימליים של תתי המערכים הרציפים של A מאורך k. כלומר עבור מערך קלט $[2, 7, 34, 6, 22, 12]$ עם $k = 3$ יוחזר

$$min\{2, 7, 34\} = 2, min\{7, 34, 6\} = 6, min\{34, 6, 22\} = 6, min\{6, 22, 12\} = 6$$

ניקוד מלא יינתן לאלגוריתם שרץ בזמן $O(nlogk)$ כאשר n הוא $|A|$.

פתרון: נשים לב כי אם נרצה לעבור על כל תתי המערכים הרציפים באורך k כל שנצטרך לעשות הוא להסתכל בכל פעם על k האיברים הראשונים וזה המערך הראשון, כלומר מאינדקס $[1, k]$ ואז לקדם את האינדקס הראשון והאחרון ב1 כלומר להסתכל על $[2, k + 1]$ וכן הלאה עד שנקבל שהגענו לאינדקס התחלה i וכן $|A| - i < k$ כלומר לא ניתן להוסיף קבוצת k־יה בסוף. כמה זה יעלה? תכף נדון בכך.

ננסה להבין מה קורה מהסיבוכיות, רוצים ממני במשך n פעמים לבצע $logk$. כלומר, שקול הדבר לבצע חיפוש בתוך k־יה ממויינת.

נראה כי אם נרצה להשיג את כל קבוצות ה"k"־יות הרלוונטיות מההסבר מעלה זה יכול להתבצע ב$O(n)$ ע"י מעבר על המערך בצורה לינארית כפי שתיארתי ביתר פירוט. כעת יש לנו את כל הקבוצות ביד. נרצה למצוא מינימום בכל אחת מהן. באופן נאיבי נוכל ממש לעבור על כל k־יה כזו, אבל זה יעלה לכל k־יה $O(k)$. כלומר פתרנו את הבעיה כעת ב$O(nk)$. נרצה לשפר זאת ־

מה שנעשה יהיה להשתמש בסעיף א', ראינו שקיים מבנה שניתן לדעת את המינימום בו ב$O(1)$! מה שנעשה יהיה בעת הכניסה למערך ליצור עץ AVL ולהכניס אליו את כל האיברים בk־יה הראשונה. זה יעלה $klogk$. כעת, בכל שלב יש לנו k־יה ביד. אנחנו נשלוף את המינימום באמצעות מבנה הנתונים שבנינו בסעיף א', ואז מה שנעשה יהיה להוציא את האיבר הראשון משמאל בk־יה (קל לדעת מי זה כי אנחנו עובדים עם מערך אז יודעים את האיבר הזה וזה די נאיבי), נוציא אותו מה שיעלה $O(logk)$ שכן מס' האיברים בכל רגע נתון בעץ יהיה k, ובמקומו נכניס את האיבר החדש שאנחנו הולכים לתחום. נשים לב שכל עוד אנחנו בסדר מבחינת טווח וההערות שאמרנו למעלה, אם היינו בטווח $[i, k]$ אזי האיבר שנוציא יהיה i, ואז עברנו לטווח $[i + 1, k + 1]$, והאיבר שנכניס יהיה $k + 1$. סה"כ מדובר בפעולת הכנסה לעץ וראינו שזה יהיה ב$O(logk)$, סה"כ בכל k־יה נוכל למצוא כך את המינימום ב$O(logk)$. מדוע? הכנסה ב$O(logk)$ והוצאה ב$O(logk)$ סה"כ $2logk = O(logk)$

כעת סה"כ סיבוכיות הפתרון: $klogk$ לבניית העץ בהתחלה, ואז במשך n פעמים במקרה הגרוע (אם $k = 1$ אזי יש לנו n פעמים כאלו) נעבור על הk־יות האפשרות ובכל אחת מהן נעבוד $O(logk)$ כמו שתיארנו ולכן סה"כ זה יעלה $nlogk$, סה"כ נקבל $nlogk + klogk = O(nlogk)$ שכן $k < n$. כנדרש.

שאלה 2:

תור עדיפויות מונוטוני הוא מבנה נתונים שתומך בפעולות הבאות:
א. $init(n)$ מאתחל תור עם המפתחות $1, .., n$. התור מאותחל פעם אחת בלבד.
ב. $Delete(x)$ מוחק את הערך x מהתור, אם x כבר מחוק (לא קיים) אזי הפעולה לא עושה דבר
ג. $DeleteMin(x)$ מוחק ומחזיר את הערך המינימלי כרגע בתור.

בנה מבנה נתונים שמממש תור עדיפויות מונוטוני. יש לתאר את המבנה, מימוש הפעולות. נתחו ומזערו לשיעורין סדרה של $n + 1$ פעולות, כאשר הפעולה הראשונה היא $init$ ולאחר סדרת n פעולות של $DELETE$ ו$DELETEMIN$. שימו לב שסדר ההוצאה של האיברים וכן האם הם יוצאים דרך פעולה $delete$ או $deleteMin$ אינו ידוע.

פתרון: אמרו לשיעורין וכן ביקשו לשיעורין ולכן אנחנו נבנה מבנה שעושה הכל ב$O(1)$, בסוף ננתח סיבוכיות לשיעורין.

נאתחל מערך A בגודל n והוא יהיה מערך בינארי. האיבר i יהיה תמיד במיקום i במערך. נגדיר 0 האיבר מחוק ו1 האיבר קיים. בעת האתחול, נגדיר $min = \infty$ ובכל שלב שנכניס איבר נשווה אל min וכך נדע את הערך המינימלי בתור. בנוסף נאתחל מצביע לאיבר הראשון בתור. הרי תמיד 1 הוא האיבר המינימלי.

פעולת $init$ תעשה מה שתיארתי מעלה.

בהינתן מחיקה, נלך לתא x ונעדכן שם במקום 1 להיות אפס. אם כבר היה אפס לא נעשה דבר. נראה כי לשיעורין, מדובר על הקצאת מטבע אחד לפעולה פשוטה $A[x] = 0$.

מחיקת מינימום: ניגש עם המצביע אל האיבר. אם יש שם 1 אזי נוכל למחוק אותו, נשנה את ערך האיבר ל0 ונקדם את המצביע. אם אין שם איבר, כלומר זה 0, המינימום כבר מחוק. אנחנו נרצה לקדם את המצביע עד שנתקל בתא עם הערך אחד. שם נמחק את האיבר המינימום כמו שרצינו, ונקדם הלאה את המצביע.

נראה ניתוח עם שיטת הבנק.
בעת אתחול הרשימה נקצה לכל איבר 3 מטבעות. אחד להכנסתו לרשימה כעת, אחד למחיקת האיבר ואחד להזזתו העתידית עם קידום המצביע. סה"כ כאן כיסינו גם את פעולת ההכנסה.
כעת לאחר הכנסת n האיברים יש בבנק $2n$ מטבעות. נשים לב שבמקרה הגרוע ביותר אותו ננתח, נזיז את המצביע n פעמים (אם למשל המצביע מאותחל לאיבר הראשון והמינימום הפך להיות בסוף). סה"כ אנחנו נשלם בכל שלב מטבע אחד להוצאה ועוד מטבע לקידום איבר, ולאחר n הפעולות הללו נשאר עם 0 איברים ביד.
לכן לשיעורין, כל פעולה חסומה ע"י 3 ולכן לשיעורין כל הפעולות יתבצעו ב$O(1)$.

שאלה 3:

נגדיר עץ בינומי משולש מדרגה k ונסמן $TB^{(k)}$ בצורה הבאה:
א. עץ בינומי משולש מדרגה 0 הוא קודקוד בודד
ב. לכל $k > 0$ עץ בינומי משולש מדרגה k הוא מורכב משלושה עצים בינומיים מדרגה $k-1$ שיסומנו $TB_1^{(k-1)}, TB_2^{(k-1)}, TB_3^{(k-1)}$. העץ בנוי בצורה בה מוסיפים את TB_1 וTB_2 כבנים של השורש TB_3. לכן למשל עבור $k = 1$ נקבל עץ עם קודקוד בודד, ולו שני בנים (1 ו2 TB המדוברים) שגם הם עם קודקוד אחד, שכן כל אחד מהם הוא מדרגה $1 - 1 = 0$ ולעץ TB^0 יש קודקוד בודד.
הוכח באינדוקציה:
1. 10 נק' ־ לכל k טבעי כולל אפס מתקיים גובה העץ $TB^{(k)}$ הוא k.
2. 15 נק' ־ הוכיחו באינדוקציה כי לכל k טבעי כולל אפס כמות הקודקודים בעץ $TB^{(k)}$ היא 3^k.

פתרון:

1. בסיס: $k = 0$, במצב כזה העץ הוא קודקוד בודד מהנתון ואכן גובה העץ של קודקוד בודד הוא אפס.
צעד: נניח שהטענה נכונה לכל עץ בינומי מסדר $k - 1$. נוכיח עבור k. יהי עץ בינומי משולש מסדר k. אזי מהנתון, הוא מורכב באופן הבא: הוא עץ מסדר $k - 1$ בעצמו, ויש לו שני בנים T_1 וT_2 שגם הם מסדר $k - 1$. נראה כי גובה $TB^{(k-1)} = k - 1$ מהנחת האינדוקציה. כעת, נשים לב (ובזמן מבחן גם נצייר על הדף להסביר ברור כי טריוואלי) שכן העץ מורכב משני תתי עצים בגובה $k - 1$, ועוד עץ כזה שכולל את השורש. ולכן המסלול הארוך ביותר שניתן לעשות יתחיל בשורש, ומשם נגיע לאחד הבנים T_1 או T_2. כמו כן, זה המסלול הכי ארוך שניתן לבצע. (די טריוואלי על דף עם ציור). ולכן, גובה העץ שמוגדר להיות המסלול הארוך ביותר מהקודקוד לאחד העלים יכלול את הקודקוד וכן את אחד מתתי העצים. גובה כל תת עץ הוא $k - 1$ וכן השורש מוסיף לגובה 1 ולכן סה"כ גובה העץ יהיה $k - 1 + 1 = k$, כנדרש.
2. בסיס: $k = 0$ אזי לעץ יש קודקוד אחד מהנתון ואכן $3^0 = 1$.
צעד: נניח שהטענה נכונה לכל עץ בינומי מסדר $k - 1$. נוכיח עבור k. נראה כי מס' הקודקודים הכולל מההגדרה כולל חיבור של שלושה עצים $TB^{(k-1)}$. אם כן, $|TB_1^{(k-1)}| = |TB_2^{(k-1)}| = |TB_3^{(k-1)}| = 3^{k-1}$ מהנחת האינדוקציה, ולכן

$$TB^{(k)} = 3^{k-1} + 3^{k-1} + 3^{k-1} = 3 * 3^{k-1} = 3^k$$

כנדרש.

שאלה 4

נגדיר k־יה אריתמטית בתור קבוצה סדורה בעלת k מספרים טבעיים כך שההפרש בין כל שני מספרים סמוכים זהה. נתון מערך בעל n מספרים שלמים שונים זה מזה. שימו לב ההפרש לא בהכרח k.
א. הצע אלגוריתם שמקבל כקלט מערך A ומס' k ומחזיר אמת אם במערך קיימת k־יה אריתמטית ושקר אחרת. האלגוריתם ירוץ בזמן $O(n^k)$. נתחו אלגוריתם בזמן הגרוע ביותר אין צורך להוכיח נכונות.
ב. הצע אלגוריתם אחר שפותר את הבעיה מהסעיף הקודם אך ירוץ בזמן $O(kn^2)$ בתוחלת. הנח שקיימת פונקציית גיבוב טובה.

פתרון:

א. זמן ריצה ממש של הפתרון הנאיבי.... בהינתן n איברים, נרצה לבדוק כמה קבוצות בגודל k קיימות. שקול ממש לבחירה ללא חזרה וחשיבות, כלומר מס' הקבוצות בגודל k יהיו $\binom{n}{k}$. נראה כי בכל קבוצה כזו שנקבל נרצה לעבור על האיברים בתוכה, ולבדוק אם אכן ההפרש בין כל שני מספרים זהים יהיה זהה. זה יעלה $O(k)$ סה"כ נקבל

$$k*\binom{n}{k} = \frac{k * n!}{k! * (n-k)!} = \frac{(n-k)! * (n-k+1) * ...... * (n-k+k)}{(k-1)! * (n-k)!} = \frac{(n-k+1) * ...... * (n-k+k)}{(k-1)!} \leq \frac{n^k}{(k-1)!} = O(n^k)$$

כנדרש.
ב. כעת אמרו תוחלת ולכן נשתמש בטבלת האש. נכניס את האיברים לתוך טבלת האש. נראה כי בהינתן n מספרים יש n^2 זוגות שנוכל ליצור מהם.

2016 מועד ב שאלה 5

צריך לכתוב תכנות דינמי שיחזיר תת סדרה פלינדרומית ארוכה ביותר.

מחרוזות במערך S כך ש$|S|=n$. נגדיר $f(i,j)$ אורך תת מחרוזת פלינדרומית ארוכה ביותר באינטרוול $[i,...j]$. $i\leq j$ תמיד. אם $i=j$ אז תמיד 1 כי באופן ריק פלינדרומי.

$$f(i,j):=\left\{\begin{array}{cc} -\infty & i>j \\ 1 & i=j \\ 2+f(i+1,j-1) & A[i]=A[j] \\ max\{f(i+1,j),f(i,j-1),f(i+1,j-1)\} & o.w \end{array}\right\}$$

ניצור מטריצה משולשית עליונה $n\times n$. נקרא לה B. בכל תא $B[i,j]=f(i,j)$, באלכסונים נמלא. (לפי תנאי נוסחת נסיגה), פתרון ב$B[1,n]$.

מקום - אם לא צריך שחזור אז זה $O(n)$ כי שומרים שני אלכסונים - נוכחי וקודם. זמן - $O(n^2)$ כל תא בזמן קבוע ויש n^2 תאים.

2015 תרגיל 5 שאלה 3

השאלה: נתונה שורה של n מטבעות שערכיהם $v_1,....,v_n$כאשר n זוגי. עוזי משחק מול שאול בתורות מתחלפים. כאשר כל שחקן פועל אך ורק בתורו. בכל תור, השחקן שזהו תורו בוחר את המטבע שנמצא בסוף השורה או את המטבע שנמצא בתחילת השורה. ומסיר אותו מהשורה לצמיתות וצובר את ערך המטבע שהסיר. הצע אלגוריתם תכנון דינמי שממקסם את הערך ששחקן יכול לצבור.

פתרון: זו שאלה מתורת המשחקים ואנחנו ננסה לגשת אליה אחרת - לא כמה אני יכול להרוויח, אלא כמה אני יכול לדפוק את שאול. המטרה שלי היא ששאול ירוויח כמה שפחות - לא שאני ארוויח כמה שיותר (וכן בעקיפין ארוויח יותר ממנו).

נגדיר את הפונקציה $f_1(i,j)$ כפונקציה שמחזירה את הערך המקסימלי עבור המטבעות $v_i,...v_j$ כאשר שחקן 1 בתור ו$f_2(i,j)$ בדומה עבור שחקן 2.

נראה כי אם נרצה לחשב את הפונקציה הזו נוכל להגדיר:

$$f_1(i,j):=\left\{\begin{array}{cc} v_i & i=j \\ \sum_{k=i}^{j}v_k-min\{f_2(i+1,j),f_2(i,j-1)\} & i>j \end{array}\right\}$$

$$f_2(i,j):=\left\{\begin{array}{cc} v_i & i=j \\ \sum_{k=i}^{j}v_k-min\{f_1(i+1,j),f_1(i,j-1)\} & i>j \end{array}\right\}$$

מדוע הפתרון הזה עובד? המטרה היא לדפוק את השני. אנחנו יודעים שסכום הרווח יהיה $\sum_i^j v_k$ של שניהם ולכן רווח של אחד זה הסכום פחות הרווח של השני. מבין האפשרויות אנחנו נרצה שהשני ירוויח כמה שפחות, מבין ההצבות בשני המיקומים האפשריים.

כעת נסתכל על סיבוכיות הפתרון. נראה ממש לא טוב. יש n^2 תאים אבל כל תא עולה לנו $O(n)$ בשל חישוב הסיגמה.

לכן נבצע עיבוד מקדים: נגדיר מערך סכומים ונקרא לו B, $B[i]=\sum_{k=1}^{i}v_k$. נראה כי נוכל לחשב מערך זה ב$O(n)$ ע"י הגדרה רקורסיבית כי $B[1]=v_1$ וכן $B[i]=B[i-1]+v_i$. נחשב את המערך הנ"ל ב$O(n)$ וכעת מילוי כל תא בפתרון הרקורסיבי יעלה $O(1)$. כעת ניצור מטריצה $n\times n$ ונמלאה לפי אלכסונים מה שיבטיח מילוי תא ב$O(1)$ ע"י שימוש בקריאות חוזרות, סה"כ נוכל להחליט גם שנשמור בכל רגע נתון רק את שני האלכסונים האחרונים והפתרון יופיע ב$[1,n]$ לכל שחקן. סה"כ סיבוכיות המקום תהיה $O(n)$ והזמן יהיה n לעיבוד מקדים ועוד n^2 סה"כ $O(n^2)$.

2015 תרגיל 5 שאלה 2

השאלה: נתבונן בוריאציה של בעיית תרמיל הגנב בשלמים. כעת נתונים שני שקים לגנב W_1,W_2 זה העלויות. שאר הנתונים הם כמו שראינו n פריטים במשקלים $w_1,...,w_n$ וערכיהם $v_1,...,v_n$והמטרה למקסם את הרווח. כתוב אלגוריתם תכנון דינמי שפותר את הבעיה.

נגדיר $f(j,b_1,b_2)$ כפונקציה שתמקסם את הרווח שניתן לעשות בלקיחת הפריטים $1,...,j$ עם התיקים במשקל b_1 וb_2 בהתאמה. כעת נגדיר את הפונקציה הבאה:

$$f(j,b_1,b_2):=\left\{\begin{array}{cc} 0 & j=0 \\ f(j-1,b_1,b_2) & w_j>b_1\wedge w_j>b_2 \\ max\{v_j+f(j-1,b_1,b_2-w_j),f(j-1,b_1,b_2)\} & w_j>b_1 \\ max\{v_j+f(j-1,b_1-w_j,b_2),f(j-1,b_1,b_2)\} & w_j>b_2 \\ max\{v_j+f(j-1,b_1-w_j,b_2),v_j+f(j-1,b_1,b_2-w_j),f(j-1,b_1,b_2)\} & o.w \end{array}\right\}$$

נראה כי זו סה"כ הכללה של המקרה ולכן נוסחת הנסיגה דומה מאוד.
כעת ניצור מטריצה מסדר גודל $n \times W_1 \times W_2$ ונמלא אותה לפי המתואר כאן בטבלה כאשר מילוי כל תא הוא $O(1)$, סה"כ נתפוס מקום בסדר גודל של $O(W_1W_2n)$ וכן זמן באותו הסיבוכיות האסימפטוטית. הפתרון יהיה בתא $[n, W_1, W_2]$.

2015 תרגיל 5 שאלה 4

נתונה קבוצה של n מספרים חיוביים המיוצגת במערך $A = \{a_1, ...a_n\}$. תת קבוצה תקרא 2־מוגבלת אם היא $A' \subseteq A$ כאשר $A' = \{a_{i_1}, ..., a_{i_k}\}$ כך ש
א. $i_j - i_{j-1} \leq 2$ עבור $2 \leq j \leq k$
ב. $i_1 = 1, i_k = n$.
ערכה של תת קבוצה 2 מוגבלת הוא $\sum_{j=1}^{k} a_{i_j}$. כתוב אלגוריתם תכנון דינמי שמוצא את ערך תת הקבוצה ה2 מוגבלת המקסימלי ביותר.

פתרון:

נתחיל מהנאיבי. יש 2^n תתי קבוצות, נבדוק לכל אחת אם מוגבלת נחשב את ערכה ונמצא את המקסימלי מה שיעלה $O(n * 2^n)$. טו מאץ'. עבור $n = 10$ נצטרך לבצע יותר מעשרת אלפים פעולות. לא בבית ספרנו בקיצור.
נכתוב פתרון רקורסיבי, נרצה להבין את הדרישות: אנחנו רוצים שההפרש בין האינדקסים השונים יהיה לכל היותר 2, כלומר או 1 או 2. שכן לא נקח אינדקס עם עצמו. ולכן ברגע שבחרנו איבר להתחיל איתו, נשים לב שיש לנו שתי אפשרויות: בהינתן שאני באינדקס i אני הגעתי אליו מאינדקס $i-1$ או אינדקס $i-2$ בלבד. כעת ננסח את נוסחת הנסיגה כדקלמן: נגדיר $f(i)$ כערך המקסימלי של קבוצה 2 מוגבלת מבין המספרים $1, ..i$. כעת נשים לב כי

$$f(i) = \left\{ \begin{array}{cc} a_1 & i = 1 \\ a_1 + a_2 & i = 2 \\ a_i + max\{f(i-1), f(i-2)\} & o.w \end{array} \right\}$$

הוכחת נכונות:
נוכיח באינדוקציה שהנוסחה מביאה ערך מקסימלי.
עבור $i = 1$ נקבל a_1 וזה הסכום המקסימלי מתוך תת הקבוצה $\{a_1\}$. עבור $i = 2$ נקבל $a_1 + a_2$ ושוב זה אכן הסכום המקסימלי מהקבוצה $\{a_1, a_2\}$.
נניח כי לכל קלט קטן מi מתקבל הפתרון המקסימלי.
יהי i כלשהו, אזי לפי הנוסחה $f(i) = a_i + max\{f(i-1), f(i-2)\}$. כעת אנחנו יודעים לפי הנחת האינדוקציה כי $f(i-1), f(i-2)$ מביאים פתרונות מקסימלים עבור תת הבעיות שלהם, ולכן אם נבחר את המקסימום מבניהם זה יהיה אכן מקסימלי ואם נוסיף לו איבר זה יגדיל את המקסימליות עבור כל i. כעת נסביר מדוע הפתרון חוקי: בהינתן אינדקס i אני חייב לקחת את אחד מהפתרונות הקודמים שלי : או את הקודם או את זה שלפניו. כמו כן אנו נשים לב שהאיבר הראשון שאני יכול להיות בו הוא 1 ואני חייב לבקר שם ולכן לסירוגין לאחריו אני אבחר מקסימלי מבין הדילוגים האפשריים. אגב, ייתכן ונבחר שלא לדלג כלל וזה בא לידי ביטוי ברקורסיה.
כעת ניצור מערך A ונמלא אותו לפי הנוסחה בצורה רקורסיבית. כעת הפתרון יהיה ב$A[n]$. סיבוכיות המקום תהיה $O(n)$ ונוכל להקטין אותה ל$O(1)$ אם נשמור בכל שלב שני ערכים קודמים וכן סיבוכיות הזמן תהיה $O(n)$.
פסודו ־

```
Algorithem(A)
creatArray(B)
B[1]=A[1]
B[2]=A[1]+A[2]
for int i=3 to n
B[i]=A[i]+max{B[i-1],B[i-2]}
return B[n]
```

2011 מועד ב שאלה 5

בבעיית הסדרה המשותפת הקצרה ביותר נתונות שתי מחרוזות T וS מאורכים m וn בהתאמה וברצוננו למצוא את האורך של תת הסדרה הקצרה ביותר R אשר מכילה את T וS כתתי סדרות. למשל, $T = adab$ ו$S = aba$ אזי $R = adbab$ היא הסדרה שכן היא מכילה את שתיהן.
פתור את הבעיה בתכנון דינמי.
פתרון: ניגש לבעיה מכיוון של לקחת ולא לקחת. נגדיר את הפונקציה $f(i, j)$ כאשר אני בסדרה S באיברים $1, ...i$ ובסדרה T באיברים $1, ..j$. הפונקציה תחזיר את הסדרה הקצרה ביותר שמכילה את האיברים משני המחרוזות. נראה כי בכל שלב נתון אנחנו נבדוק את התנאי הבא: אם $T[j] = S[i]$ נוכל לבחור לקחת את התו המשותף. אחרת, נוכל לבחור גם שלהתקדם. ננסח זאת פורמלית כך:

$$f(i,j)=\left\{\begin{array}{cc} 0 & i=0\wedge j=0 \\ 1+f(i-1,j-1) & T[j]=S[i] \\ min\{f(i,j-1)+1,f(i-1,j)+1\} & o.w \\ j & i=0\wedge j>0 \\ i & j=0\wedge i>0 \end{array}\right\}$$

כעת ניגש לפתרון התכנון הדינמי. נבנה מטריצה B בגודל $n\times m$ ונאחסן בתאיה $B[i,j]=f(i,j)$. נמלא אותה לפי עמודות שכן ברגע נתון אני זקוק לתא שמעליי בשמאל למעלה, משמאלי ומעליי. זה יתאים בדיוק לעמודות מלמעלה למטה. כך מילוי כל תא יעלה $O(1)$, כעת נמלא לפי תנאי נוסחת הנסיגה שמפורטים מעלה. הפתרון יהיה כמובן ב$B[n,m]$. סיבוכיות הזמן תהיה $O(nm)$ וגם המקום. נראה כי אם נשמור בכל שלב שני עמודות בלבד נוכל להוריד את סיבוכיות המקום להיות $O(n)$. כעת נראה פסודו -

```
Algo(T,S)
creat-Metrix-B(n×m)
for int i=0 to n
for int j=0 to m
if i =0 and j=0 B[i][j]=0
if i=0 and j>0 so B[i][j]=j
if j=0 and i>0 so B[i][j]=i
if T[j]=S[i] so B[i][j]=1+B[i-1,j-1]
else
B[i,j]=min{B[i,j-1]+1,B[i-1,j]+1}
```

אלגו 1 2012 שאלה 3 אלגו

השאלה: תארו כיצד ניתן למיין n מספרים שלמים, כל $x\in[0,n^2-1]$. מה סיבוכיות זמן הריצה?

פתרון: למיין תמיד נוכל ב$O(nlogn)$ ולכן למעשה נרצה למצוא שיטת מיון בה נמיין ב$O(n)$

אנחנו יודעים שאם נשתמש במיון בסיס נקבל $O(d(n+R))$ כאשר R הוא הבסיס וd הוא מס' הספרות במספר. בואו נבחר את הבסיס להיות $R=n$, נקבל

$$d=log_n(n^2-1)$$

אלגו 1 2018 שאלה 2 תרגיל 6

השאלהך נתון מערך A באורך n של מספרים שלמים שערכם יכול להיות חיובי שלילי או אפס. תת מערך $A[i,..j]$ של A נקרא מקטע חיובי אם סכום המספרים בתת המערך הזה גדול מאפס. כתבו אלגוריתם חמדני שמוצא את מס' המינימלי הדרוש של מקטעים חיוביים כך שכל מספר חיובי במערך יכלל באחד מהם והוכח נכונותו. כלומר: כתוב בשני שורות את האלגוריתם, והוכח את

למה 1. למת הבחירה החמדנית

למה 2. תכונת תת המבנה האופטימלי

למשל, עבור המערך $A=[\mathbf{2,-1,3,-4,1},-9,\mathbf{2,-1,3},-8,\mathbf{5,-2,1}]$ יוחזר 3 (המערכים המודגשים הם אלו שמשתמשים בהם)

פתרון:

מהדוגמה הבנו היטב את השאלה. צריך למצוא מס' מינימלי של תתי מערכים רציפים של A כך שסכומם חיובי. וכן מספרים שליליים לא חייבים להכלל בו.

הפתרון יהיה להלן: אנחנו נסרוק את המערך ונסכום את המספרים שנהיה בו. כלומר, לפני כל שלב שנחליט לצרף את המספר אל תת המערך שלנו אנחנו נסתכל על הערך שלו. בהינתן שאני עם תת מערך $[i,...j]$ עם משקל v_i נניח. אם האיבר ה$j+1$ שאני רוצה לבקר בו הוא כזה שמקיים $v_i+A[j+1]<0$ אנחנו נבחר שלא לקחת אותו. כלומר, אם הוא איבר שלא יזיק לחיוביות שלי במערך נקח אותו. אם הוא יוריד את סכום המערך שלי לשלילי אנחנו נקטע את תת המערך, נגדיל באחד את מס' תתי המערכים, נדלג על האיבר הבא כי הוא שלילי ונמשיך הלאה. נשים לב שאם גם ההבא יהיה שלילי גם אותו לא נהיה חייבים לקחת, אלא נפתח מערך חדש ברגע שנתקל במספר חיובי. סיבוכיות פתרון זה תהיה $O(n)$ כמעבר לינארי על איברי המערך.

כעת ניגש להוכיח את שני הלמות

למה ראשונה: למת הבחירה החמדנית - קיים פתרון אופטימלי לבעיית המספרים בו כל מקטע הוא מקסימלי.

יהי פתרון אופטימלי OPT לבעיה. נניח שיש בו מקטע $[i,j]$ שאינו מקסימלי.

מקרה ראשון - ניתן להאריכו ימינה אם $A[j+1]$ קיים וכן $sum[I,j]+A[j+1]>0$, נגדיר $OPT'=[i,j+1]$.

מקרה שני - ניתן להאריכו שמאלה, כלומר קיים $A[i-1]$ וכן $A[i-1]+Sum[i,j]>0$. נגדיר $OPT'=[i-1,j]$

בשני המקרים OPT חוקי וכן $OPT' \leq OPT$ בגודל שכן מדובר באותו מס' קטעים לפחות אם לא פחות מכך. לכן OPT' אופטימלי.

למה שנייה: פתרון שיורכב מבחירה של המקטע הראשון כך שיהיה מקסימלי, בשילוב פתרון לתת הבעיה שכוללת את המקטעים הבאים הוא פתרון אופטימלי לבעיה.

הוכחה: נב"ש שA אינו אופטימלי וקיים B כך ש$|A| < |B|$. יהי $[1,k]$ המקטע הראשון בA.

א. אם B מתחיל גם במקטע $[1,k]$ אזי מהנחת האופטימליות הן A והן B פותרים תת בעיה $[k+1,n]$, בסתירה ל$|A| > |B|$

ב. אם B מתחיל במקטע קצר יותר $j < k$ כך ש$[1,j]$. אזי, מהלמה הראשונה היינו מאריכים את B ל$[1,k]$ ומקבלים פתרון אופטימלי יותר בסתירה.

ג. אם B לא חוקי = בפרט סתירה.

סה"כ לא יתכן אלגוריתם אופטימלי יותר.

שמואל קליין מבחן 2020

שאלה 1: כתוב אלגוריתם שמקבל כקלט מצביע לשורש עץ בינארי כמעט שלם T ומחזיר כפלט את מס' הצמתים d שיש להוסיף לעץ ברמה התחתונה על מנת שיהיה עץ שלם.

ניקוד מלא ינתן לאלגוריתם שזמן הריצה שלו הוא $O(h^2)$ כאשר h הוא גובה העץ, ניקוד חלקי לאלגוריתם שזמן ריצתו הוא $O(min\{n,(d+1)h\})$ וניקוד שזמן ריצתו $O(n)$ גם כן יקבל נקודות, כאשר n הוא מס' הצמתים כרגע בעץ, אך פחות מהמקרים שפורטו מעלה. הנח שלכל צומת בעץ יש רק 3 שדות : מפתח (אין חשיבות למידע זה לבעיה שלנו), מצביע לבן ימני ומצביע לבן שמאלי.

פתרון:

נתחיל מהניקוד הנמוך לגבוה, וננסה לפתור בכל הדרכים (כי זה תרגול אז למה לא?)

אנחנו יודעים שעץ בינארי כמעט שלם הוא עץ מאוזן (ראינו זאת כשדנו בערימה), ולכן אנחנו יודעים שגובהו יהיה $h = O(logn)$. כעת,

א. פתרון בזמן לינארי, אנחנו יכולים לסרוק את העץ באמצעות סריקת $in - order$ וכך לחשב את מס' האיברים כעת בעץ. בעץ בינארי שלם יש $2^{h+1} - 1$ איברים. וכן אם נסרוק גם נוכל למצוא את גובה העץ ב$O(logn)$, ולכן סה"כ אם מצאנו את גובה העץ אנחנו יודעים שבעץ בינארי שלם של העץ הזה אמורים להיות $2^{h+1} - 1$ קודקודים. נסמן ערך זה כx. מהסריקה $in - order$ מצאנו שיש בעץ כרגע y איברים. ולכן סה"כ נחזיר את $d = x - y$ מס' האיברים שיש להוסיף. סה"כ עבדנו $n + logn = O(n)$.

ב. נעשה ריצה במקביל של הפתרון מהסעיף הקודם, וכן נבצע $d + 1$ פעמים ירידה משורש העץ אל אחד העלים. אנחנו יודעים שנצטרך לרדת למטה $d + 1$ פעמים וכן גובה העץ הוא $logn$, ולכן לאחר $d + 1 * h$ פעולות אנחנו נוכל לדעת כמה איברים עלינו להשלים, שכמובן נרוץ במקביל לפתרון של סעיף א' וכאשר פתרון אחד ימצא את d נסיים.

ג. כעת נרצה למצוא אלגוריתם שזמן ריצתו הוא $O(h^2)$. כלומר אם מס' האיברים שלי כרגע בעץ הוא n, אנחנו יודעים כפי שהסברתי כי $h = logn$, ולכן הפתרון צריך להתבצע ב$O(log^2 n)$.

נקודת המפנה היא שבעץ כמעט שלם ישנה נקודת מפנה ת ממלאים אותו משמאל לימין ויש נקודה שמשמאל לה הכל מלא ומימין לה הכל ריק. איך נמצא אותה? נרצה לחפש בינארית ובכל שלב לבטל חצי מהאיברים. נאתחל מונה $counter = 0$. נרד לבן הימני של השורש ונוסיף 1. נרד לבן השמאלי של השורש כל עוד אין בנים שמאליים, ובכל ירידה נגדיל את הקאונטר באחד. כעת אם $counter = h$, אזי האיבר שהוא אחד ימין לאמצעי מהעלים קיים, ומהגדרת עץ בינארי כמעט שלם גם כל האיברים בעלים בתת העץ משמאל קיימים. לכן אנחנו נסתכל רק על החצי הימני של העץ, יענו נפעיל אלגוריתם שוב על בן ימני של השורש.

אם $counter = h - 1$ אזי כל העלים שבחצי הימני של העץ ברמה התחתונה לא קיימים ולכן נסתכל רק על החצי השמאלי (נסתכל במצב זה על בן שמאלי של שורש וכך נפעיל ברקורסיה)

כלומר ממש הפרד ומשול בו בכל שלב אנחנו מעיפים חצי.

כעת בכל פעם שנפעיל את האלגוריתם נתחזק מונה שנקרא לו x ונאתחלו לn, אם קיבלנו באלגוריתם $counter = h$ אזי נגדיר $x - = \frac{n}{2}$ כי יודעים שלפחות חצי קיימים. אם קיבלנו $counter = h - 1$ גם נחסר חצי כי לא קיימים. סה"כ בסוף נחזיר את x, מדוע אותחל לn? כיוון שברמה האחרונה n איברים.

סיבוכיות: כל פעולה תקח כגובה העץ, $O(h)$ כי יורדים במקרה הגרוע h רמות. נפעיל את האלגוריתם בכל פעם ונרד רמה אחת והוא יופעל על h רמות ולכן $O(h^2)$. אם נרצה ממש לדייק בכל שלב אנחנו חותכים חצי בחיפוש בינארי, ממה אנחנו חותכים חצי? על רמות העץ ולכן יש h איטרציות. כמו כן בכל איטרציה אנחנו יורדים כגובה העץ ולכן $O(h^2)$.

שאלה 2:

להלן תיאור בעיית אורך תת הסדרה המשותפת המשולשת הארוכה ביותר:

קלט: שלוש סדרות X, Y, Z באורך n.

פלט: אורך הסדרה A שהיא תת סדרה משותפת ארוכה ביותר של שלושת הסדרות.

פתור את הבעיה בתכנון דינמי.

פתרון:

נגדיר $f(i,j,k)$ כאורך תת הסדרה המשותפת המקסימלית עבור איברים $1,...i$ בX ו$1,...j$ בY ו$1,....k$ בZ. נשים לב שבכל שלב יש לנו שתי אפשרויות: אם $X[i]=Y[j]=Z[k]$ ניקח את האיבר ונגדל את אורך תת הסדרה ב1 ונצמצם את הקלט להיות $f(i-1,j-1,k-1)$, אחרת נבחר אופטימזציה מבין האפשרויות להורדת תו מהסדרות. נתאר זאת פורמלית כדקלמן:

$$f(i,j,k) := \left\{ \begin{array}{cc} 0 & i \vee j \vee k = 0 \\ 1+f(i-1,j-1,k-1) & X[i]=Y[j]=Z[k] \\ max\{f(i,j,k-1), f(i,j-1,k), f(i-1,j,k)\} & o.w \end{array} \right\}$$

כעת ניצור מטריצה B מגודל $n \times n \times n$ ונמלא אותה באלכסונים תלת מימדיים כך שמילוי כל תא יקח $O(1)$, הפתרון יהיה בתא $B[n,n,n]$, כל תא $B[i,j,k]=f(i,j,k)$. סה"כ נוכל גם להוריד את סיבוכיות המקום ל$O(n^2)$ ע"י שמירת אלכסון רלוונטי וכן סיבוכיות זמן תהיה $O(n^3)$.

שאלה 3:

הצע מבנה נתונים מחסנית מינימום התומך בפעולות המחסנית הרגילות ובנוסף המבנה יתמוך בפעולה **החזרת איבר המינימום ללא הוצאה מהמבנה.**

סיבוכיות זמן אופטימלית נדרשת וכן סיבוכיות מקום לינארי במס' האיברים במחסנית.

פתרון:

נשתמש במחסנית עזר ונקרא לה B ולמחסנית שלנו נקרא A. הרעיון יהיה פשוט ־ בכל שלב אנחנו עומדים להכניס את איבר המינימום למחסנית העזר שלנו. אפרט כעת:

נבנה מחסנית עזר בשם B והיא תהיה מחסנית המינימום. נקבל את האיבר a_1. הוא תמיד יכנס למחסנית המינימומים. כעת נקבל a_2. נבדוק האם נכון לרגע זה הוא המינימום במחסנית. כיצד? נגדיר $B.push(min\{a_1,a_2\})$ כלומר אנחנו נדחוף אותו שוב גם אם היה כבר במחסנית. ובאופן כללי יותר אם בראש מחסנית B כרגע יש a_i וקיבלתי a_jנבצע $B.push(min\{a_i,a_j\})$. סה"כ זו תהיה פעולת הכנסה, נכניס את האיבר למחסנית A ונבדוק את המינימום הלוקאלי.

כעת נדון בפעולת הוצאה, נוציא רגיל מהמבנה וכן נמשוך את איבר המינימום שכעת בראש המבנה B. כיוון שזה היה המינימום נכון לשעתו של A כאשר בראשה היה האיבר a שהרגע הוצאנו.

זה הכל ־ כעת ננתח סיבוכיות:

מקום ־ בכל שלב אם במחסנית A יש k איברים גם במחסנית B יש k איברים וסיבוכיות המקום לינארית.

זמן ־

ננתח זאת לפי שיטת הבנק: לכל איבר נעניק 4 מטבעות: אחד לשימוש מיידי להכנסה, אחד להכנסתו אל מחסנית המינימומים, אחד להוצאתו מהמחסנית בהמשך ואחד להוצאת האיבר המינימום שהיה נכון לשעתו בB. סה"כ הכנסה נשלם 4 מטבעות ובהוצאה לא נשלם כלום ־ שילמנו על כך מראש, סה"כ כל פעולה חסומה ב4 כלומר $c_i \le 4$ ולכן סדרת פעולות $T(n) \le 4n$ ולשיעורין $\frac{T(n)}{n} \le 4 = O(1)$.

2005 מועד ב טומי

שאלה 3:

נתונה רשימה של n מספרים. ברצוננו לבצע את הלולאה הבאה n פעמים.

א. הוצא את שני המספרים הקטנים ביותר x,y מהרשימה והדפס אותם

ב. צור מס' חדש $z=x+y$

ג. הכנס את z לרשימה

תאר אלגוריתם שיבצע זאת בזמן $O(nlogn)$

פתרון:

א. נבנה ערימה $O(n)$, נוציא שניים קטנים $2logn$, נכניס איבר $logn$. סה"כ כפול n פעמים $3nlogn = O(nlogn)$.

שאלה 6

נתונה קבוצה של n מספרים שברצוננו למיין וידוע שיש רק $O(log^2n)$ מספרים שונים בקבוצה. תאר אלגוריתם למיון הקבוצה שיעבוד בזמן ומקום $O(n)$.

פתרון:

$n-log^2n$ חופפים.

בוא נכניס לטבלת האש, בכל פעם נבדוק האם האיבר קיים. אם קיים נגדיל את הקאונטר שלו את מס' המופעים שלו ואם הוא לא קיים כן נכניס אותו.זה עלה לנו $O(n)$.

עכשיו נכניס את האיברים למערך B בגודל log^2n. נמיין אותו

$$log^2nlog(log^2n) < log^2n < n = O(n)$$

ואז נעבור באופן לינארי על מערך חדש C עם דופליקטטס וסיימנו בזמן לינארי.

2019 טכניון מועד ב סמסטר ב שאלה 3

ארנב מגיע לבית אריזה לגזר ומוצא שורה של n ארגזים ממוספרים $1,...,n$ ומלאים בגזרים. הארנב ממהר לבחינו באלגוריתמים ויכול להתעכב בבית האריזה T דקות לכל היותר.

הארנב יכול לדלג על ארגז, או להתעכב בו $1,2,3$ דקות בדיוק. לכל ארגז $1 \leq i \leq n$ ולכל $j \in \{1,2,3\}$ ידוע כי $C(i,j)$־ כמה גזרים יכול לאסוף הארנב מהארגז הi תוך j דקות.

לכל $1 \leq i \leq n$ מתקיים $c(i,1) \leq c(i,2) \leq c(i,3)$. הצע אלגוריתם תכנון דינמי שמחשב את מס' הגזרים המקסימלי שהארנב יכול לאסוף בביקורו בבית האריזה בסיבוכיות $O(nT)$

פתרון:

המטרה למקסם את מס' הגזרים שיוכל לאכול הארנב. נגדיר $f(i,j)$ כאשר $1 \leq j \leq T$ וכן $1 \leq i \leq n$. הפונקציה תחזיר את מס' הגזרים המקסימלי שהארנב יכול לאכול בהינתן j דקות ובהינתן הארגזים $1,...i$. כמובן שהפתרון לבעיה יהיה $f(n,T)$ שייוצג במטריצה ממימדים אלו. זה גם יהיה דרישת סיבוכיות הזמן והמקום ולכן נרצה למלא כל תא ב$O(1)$ זמן. כעת נגדיר את נוסחת הנסיגה הבאה:

$$f(i,j) := \left\{ \begin{array}{cc} 0 & j=0 \vee i=0 \\ max\{f(i-1,j), f(i-1,j-1)+c(i,1)\} & j=1 \\ max\{f(i-1,j), f(i-1,j-1)+c(i,1), f(i-1,j-2)+c(i,2)\} & j=2 \\ max\{f(i-1,j), f(i-1,j-1)+c(i,1), f(i-1,j-2)+c(i,2), f(i-1,j-3)+c(i,3)\} & j \geq 3 \end{array} \right\}$$

הסבר נכונות:

עבור $j=1$ דקות לארנב יש שתי אפשריות בדיוק ־ או להתעכב בכל הדקה ולאכול או לדלג לתחנה הבאה (נניח שלוקח אפס שניות לדלג מתחנה לתחנה.....)

עבור $j=2$ זה 3 אפשריות ־ לדלג, להשאר דקה או שניים.

עבור $j=3$ ומעלה זה לבחור את האופטימיזציה, כיוון שמותר "לעשות הכל".

כעת ניצור מטריצה בגודל $n \times T$ ונמלא אותה לפי נוסחת הנסיגה, נרצה למלא לפי העמודות כי אני תמיד זקוק לתאים מעליי או מעמודה שמאלית שלי. סה"כ זמן מילוי לתא יהיה $O(1)$ ולכן סיבוכיות הזמן והמקום תהיה $O(nT)$. נראה כי ניתן להוריד סיבוכיות מקום ל$O(n)$ אם נשמור בכל שלב שני עמודות אחרונות בלבד.

2019 טכניון אלגו שאלה 1 ב

ישנן n תמונות בתערוכה המפוזרות לאורך מסדרון חד מימדי. התמונות מפוזרות בנקודות $x_1 < x_2 < ..... < x_n$. אנחנו מעוניינים להציב שומרים בתערוכה. כל שומר מכסה רדיוס של 1 מסביבו. כלומר, אם הוא ניצב בנקודה $y \in \mathbb{R}$ אזי הוא משגיח על התמונות בקטע $[y-1, y+1]$. ניתן להציב שומר בכל נקודה לאורך המסדרון.

כתוב אלגוריתם חמדן (בכמה שורות) שרץ בזמן $O(n)$ ומחשב את מס' השומרים המינימלי שיש להציב ככה שלכל תמונה יהיה לפחות שומר אחד שמשגיח עליה.

פתרון: נתחיל מהתמונה הראשונה וכל עוד יש תמונות שאינן מכוסות נציב שומר בנקודה x_i+1, נדלג על כל הנקודות x_j שהשומר הזה מכסה עד שנגיע לתמונה הבאה שלא מכוסה. סה"כ נעבור פעם אחת בצורה לינארית על המערך וזה יצא $O(n)$.

כעת נוכיח נכונות.

למה 1. למת הבחירה החמדנית: קיימם פתרון אופטימלי לבחירה חמדנית בה נבחר בכל פעם לשים שומר בנקודה הראשונה השמאלית הלא מכוסה בשומריסץ

הוכחה: יהי x_i הנקודה הראשונה הכי שמאלית הלא מכוסה ויהי פתרון OPT. אם $x_i \in OPT$ סיימנו. אחרת, x_i לא בOPT. כלומר בהכרח בשביל שהפתרון יהיה חוקי קיימת x_j במרחק מקסימלי של 1 מx_i שמכסה את הקטע. נבנה פתרון חדש: $OPT' = OPT/\{x_j\} \cup \{x_i\}$. מתקיים $|OPT'| = |OPT| - 1 + 1 \leq |OPT|$ ואופטימלי. חוקי גם כיוון שכל התמונות שהשומר הישן כיסה מכוסות כי הצבנו נקודה במרחק קטן מ1 וייתכן שהשומר החדש מכסה תמונות נוספות מימין.

למה2. תכונת תת המבנה האופטימלי: בחירה חמדנית כמתואר מעלה בתוספת פתרון אופטימלי לשאר התמונות לאחר התמונה השמאלית ביותר הלא מכוסה הוא פתרון אופטימלי לבעיה.

הוכחה: יהי אלגוריתם אופטימלי A. נב"ש שאינו אופטימלי כלומר קיים פתרון B כך ש$|A| < |B|$. אזי מההגדרה $x_i \in A$ וכן $x_i \in B$. בפרט $A/\{x_i\}$ פתרון אופטימלי לתת הבעיה מהנתון. ולכן בהכרח

$$|A/\{x_i\}| \leq |B/\{x_i\}| \longrightarrow |A|-1 \leq |B|-1 \longrightarrow |A| \leq |B|$$

בסתירה.

2016 טומי מועד א' שאלה 1

באלגוריתם סלקט המקורי העפנו $\frac{3n}{10}$ מהאיברים בכל שלב ע"י חלוקה לחמישיות.

כעת נתבונן בהצעה הבאה: נבחר את איבר החלוקה K כחציון של תת קבוצה אחרת של A למשל $k = select(A[1:\frac{n}{2}], \frac{n}{4})$ והיינו מתקבלים ששתי הקבוצות שמתקבלות מהחלוקה הם $S_1 = \{x \in A | x < k\}$ ו$S_2 = \{x \in A | x > k\}$ מקיימות $max\{|S_1|, |S_2|\} \leq \frac{3}{4}n$. מה רע בהצעה זו?
פתרון:
הם בחרו במקום לבחור את חציון החציונים לבחור את האיבר ה$\frac{n}{4}$ בגודלו בצד הימני מה שיגרור לנו למעשה שנשאר עם $\frac{3n}{4}$ איברים. כמו כן אנחנו מבצעים קריאה בגודל $\frac{n}{2}$ למציאת הk הדרוש. ולכן נקבל

$$T(n) = T(\frac{n}{2}) + T(\frac{3n}{4}) + O(n)$$

אם ננסה להוכיח באינדוקציה נקבל בסיס נכון, אך בצעד:

$$\frac{cn}{2} + \frac{3cn}{4} + n < cn$$

כלומר

$$1.25cn + n < cn \to 1.25c + 1 < 1 \to c < 0$$

בסתירה.
2016 טומי מועד א שאלה 5
נתון מערך דו ממדי $A[1, ...n, 1...., m]$ $n \times m$ שכל שורה ועמודה בו ממויינות בסדר לא יורד כלומר לכל $1 \leq i \leq n, 1 \leq j \leq m$ מתקיים $A[i,j] \leq A[i+1,j]$ וגם $A[i,j] \leq A[i,j+1]$. כלומר במילים אחרות - העמודות ממויינות וכן השורות ממויינות. עליכם בהינתן מספר k להציע אלגוריתם שמוצא תא במערך A שמכיל את הערך k ומחזיר אינדקסים (i,j) שלו או מחזיר שאין תא כזה. זמן האלגוריתם יהיה $O(n+m)$
פתרון: נסתכל על האיבר $A[1,m]$. כעת נבדוק אם $A[i,j] > k$ נרצה לזוז שמאלה תא אחד במערך (כי בהכרח הוא יהיה לפני. אם $A[i,j] < k$ אזי הוא עתיד להיות באחד מהשורות למטה באותה עמודה ולכן רד למטה, אחרת $A[i,j] = k$ מצאת את k.

2016 טומי מועד ב'

שאלה 2: עץ טרינארי מלא הוא עץ טרינארי שבו לכל קודקוד פנימי יש 3 ילדים בדיוק. הוכח באינדוקציה על מבנה העץ כי לעץ טרינארי מלא עם n עלים יש $\frac{n-1}{2}$ קודקודים פנימיים.
פתרון:
נוכיח באינדוקציה על n.
בסיס: $n = 1$ אזי יש לי בן יחיד שהוא עלה ואין עלים פנימיים כלומר יש 0 עלים פנימיים ואכן $0 = \frac{1-1}{2}$.
צעד: נניח נכונות לכל עץ עם $n' < n$ עלים.
יהי עץ טרינארי T. נסמן את בניו T_R, T_M, T_L. בהכרח לכל אחד מהעצים יהיה לפחות עלה אחד ולכן כל אחד מהעצים מקיים שמס' העלים בו קטן מn. כלומר סה"כ שלושתם מקיימים את הנחת האינדוקציה.
נסמן את מס' העלים בעצים בהתאמה n_1, n_2, n_3. וכן מס' העלים הכולל בעץ יקיים $n = \sum_{i=1}^{3} n_i$. כעת מתקיים כי מס' הקודקודים הפנימיים הכולל בעץ יהיה:

$$\frac{n_1 - 1}{2} + \frac{n_2 - 1}{2} + \frac{n_3 - 1}{2} + 1 = \frac{n-3}{2} + 1 = \frac{n-1}{2}$$

כנדרש.
שאלה 3:
נתונות n סדרות ממוינות $A_1, ...A_n$ של$I_1, ..., I_n$ איברים בהתאמה. ברצוננו למזג אותן לסדרה אחת ממוינת. נניח שצריך $a+b$ השוואות כדי למזג סדרה באורך a עם סדרה באורך b. אם $n = 2$ יש רק אפשרות אחת למיזוג, אם $n = 3$ יש אפשרות למזג $A_1(A_2A_3)$ או $(A_1A_2)A_3$ ומס' ההשוואות יהיה $(I_2 + I_3) + (I_1 + I_2 + I_3)$ או $(I_1 + I_2) + (I_1 + I_2 + I_3)$ אם $n = 4$ יש כבר 5 אפשרויות.
א. מה הקשר בין מס' האפשרויות למזג n סדרות ממוינות לבין עצים בינאריים?
ב. כמה אפשרויות יש למיין 10 זוגות?

תשובה:

א. נראה כי מס' האפשרויות למזגן שקול למס' העצים הבינאריים עם $n-1$ קודקודים.

ב. יש מס' קטלן ה־n־י אפשרויות ולכן $C_9 = \frac{1}{10}\binom{18}{9} = \frac{18!}{10!} = 11*12*....*18$.

שאלה 4:

נתון מערך $A[1,...n]$ כך שנתון שקיים אינדקס $1 \le k \le n$ עבורו מתקיים:

$$A[1] < A[2] < .... < A[k] > A[k+1] > ..... > A[n]$$

א. מצא את k בזמן $O(logn)$

ב. מצא את k בזמן $O(logk)$

פתרון:

א ־ זמן הנדרש שקול לנוסחת הנסיגה הבאה $T(n) = T(\frac{n}{2}) + O(1)$, כלומר, חיפוש בינארי. מה שנעשה בכל שלב יהיה להסתכל על החציון במערך. נסמן $j = \frac{n}{2}$, כעת אם מתקיים $A[j-1] < A[j]$ אזי השבירה בה k נמצא עוד לא קרתה, היא תקרה בצד הימני ולכן נלך לבצע אלגוריתם על צד ימין. אחרת, נלך לצד שמאל. סה"כ לינארי וכמובן $O(logn)$.

ב ־ כעת נרצה לבצע חיפוש אקספוננציאלי. נקפוץ בקפיצות של 2 בכל פעם כלומר $1, 2, 4, 8, 16.....$. בכל שלב נהיה באינדקס i. אם נקבל $A[i-1] < A[i]$ השבירה עוד לא אירתה. כעת נניח שנגיע לאינדקס j עבורו $A[j] > A[j-1]$ אנחנו יודעים שבאינדקס הקודם לא הייתה שבירה ולכן נותר לחפש ב־2^i איברים סה"כ נקבל $2logk = O(logk)$.

שאלה אקראית חשובה מהאינטרנט

נגדיר את הסדרה הבאה:

$$g_n := \left\{ \begin{array}{cc} 1 & n = 1, 2, 3 \\ g_{n-1} + g_{n-2} + g_{n-3} & n \ge 4 \end{array} \right\}$$

א. פתרו את הבעיה בתכנון דינמי בהינתן $n \in \mathbb{N}$ מחשב את g_n.

ב. שפר סיבוכיות בעזרת פתרון אחר.

פתרון:

נגדיר את הפונקציה g_n לפי נוסחת הנסיגה כאשר $g(i)$ יהיה הפלט של האיבר ה־g_i בסדרה. ניצור מערך בגודל n ונמלא כל תא בו לפי נוסחת הנסיגה. נראה כי מילוי כל תא אורך $O(1)$ וכן אין צורך לשמור את האיברים פרט לשלושת האיברים האחרונים אם בא לנו להוריד את המקום ל־$O(1)$, בכל מקרה הפתרון יהיה ב־$O(n)$ ע"י הפסודו הבא:

```
Algo(A)
createArray(B[1,...n])
B[1]=B[2]=B[3]=1
for int i=4 to n:
B[i]=B[i-1]+B[i-2]+B[i-3]
return B[n]
```

נרצה לשפר ולכן נעזוב את התכנון הדינמי. נשים לב כי

$$g_n = g_{n-1} + g_{n-2} + g_{n-3}$$

$$g_{n-1} = 1g_{n-1} + 0g_{n-2} + 0g_{n-3}$$

$$g_{n-2} = 0g_{n-1} + 1g_{n-2} + 0g_{n-3}$$

כלומר

$$\begin{pmatrix} g_n \\ g_{n-1} \\ g_{n-2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} g_{n-1} \\ g_{n-2} \\ g_{n-3} \end{pmatrix}$$

כעת נשים לב כי לכל i יתקיים

$$\begin{pmatrix} g_n \\ g_{n-1} \\ g_{n-2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}^i \begin{pmatrix} g_{n-1-i} \\ g_{n-2-i} \\ g_{n-3-i} \end{pmatrix}$$

נרצה כי $n-3-i \geq 1$ ולכן $n-4 \geq i$ הוא הערך המקסימלי כלומר

$$\begin{pmatrix} g_n \\ g_{n-1} \\ g_{n-2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}^{n-4} \begin{pmatrix} g_3 \\ g_2 \\ g_1 \end{pmatrix}$$

$$\begin{pmatrix} g_n \\ g_{n-1} \\ g_{n-2} \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}^{n-4} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

כעת קיבלנו כי בהכפלת מטריצות (העלאה בחזקת - $O(logn)$) ולאחריהן הכפלה בוקטור ($O(1)$) נקבל את g_n כנדרש ב$O(logn)$. שיפור משמעותי לעומת $O(n)$.

2024 מועד ב' גלעד וטליה

שאלה 1:

ממשו מבנה נתונים לשמירת נקודות (x,y) במישור הממשי שתומך בנקודות הבאות:

א. $indert(x,y)$ מכניס את הנקודה (x,y) למבנה ביעילות $O(logn)$ כאשר n הוא מס' הנקודות במבנה

ב. $Delete(x,y)$ מסיר נקודה מהמבנה ביעילות $O(logn)$

ג. $sameFactor(a)$ מקבלת כקלט מס' ממשי a ומדפיס את כל הנקודות במבנה המקיימות $a=xy$. היעילות הנדרשת הינה $O(k+logn)$ כאשר k הוא אורך התשובה.

פתרון:

נשתמש בעץ AVL. השאילתא היחידה שנראית בעייתית היא האחרונה. נראה כי נרצה לשמור את העץ לפי שדה מסויים. נשים לב כי נוכל לשמור את השדה להיות xy וכן לשמור בכל קודקוד את הערכים x,y שהובילו למכפלה הרלוונטית.

כעת אם נרצה להכניס לעץ, נכניס את המכפלה xy בתור מפתח ואת ערכי הנקודה בתור שדות ונכניס לעץ AVL רגיל וזה אכן $O(logn)$

אם נרצה למחוק, מה שנעשה יהיה קודם כל לחפש איברים שיביאו למכפלה xy. כעת אם לא מצאתי בסדר נחזיר לו שלא יתכן. אבל אם מצאתי, ייתכן שמצאתי כמה. נשים לב כי ייתכן שנגיע למכפלה מסויימת משני מקומות שונים למשל $(1,3)$ ו$(3,1)$ וזה נקודות שונות אבל נגיע לערך זהה עבורם ואז נתקל בבעיה מבחינת היחודיות של המפתח.

לכן מה שנעשה יהיה ליצור עץ AVL עבור המפתחות xy. לכל אחד מהמפתחות יהיה מצביע לעץ AVL משלו. כל אחד מהמפתחות יחזיק פוינטר לעץ AVL שיכיל את כל הנקודות (x,y) שמכפלתן היא xy.

כעת למימוש, הכנסת נקודה תהיה לפי ערך הxy. נכניס אותה ואם מצאנו שכבר קיימת כזו, אזי לא נוסיף לעץ הגדול אלא נוסיף את הנקודה לתוך העץ שמאוחסן בפנים עבור ערכי הנקודות שמקיימות xy. גודל העץ הפנימי הוא $logw$ כאשר w הוא מס' האיברים בעץ הפנימי ובהכרח $w \leq n$ ולכן חיפוש בעץ הגדול יעלה $O(logn)$ ובקטן $O(logw) = O(logn)$ ולהסרה וסה"כ $O(logn)$. למחיקה בדומה. אם מצאנו איבר בו מחקנו את הנקודה האחרונה כלומר יש xy שמאחסן עץ avl עם נקודה אחת, ומחקנו אותה, אזי נמחק גם את ה$node$ של xy מהעץ הגדול.

כעת ל$sameFactor$, נבצע חיפוש בעץ למצוא את הנקודה xy בעץ הגדול, אם לא מצאנו נחזיר ביי. אחרת מצאנו, נבצע הדפסה של כל הנקודות שבתוך העץ הזה. נראה כי בהינתן שיש בתוך העץ הזה k איברים יעלה להדפיס אותם בסריקה $O(k)$ וסה"כ $k + logn$ וסיימנו.

שאלה 2:
נגדיר מבנה נתונים S שתומך בפעולות הבאות:
א. $insert(x)$ מכניס את האיבר x לתוך מבנה הנתונים
ב. $DeleteLargestHalf()$ מוחק את $|S|/2$ האיברים הגדולים ביותר שנמצאים כרגע במבנה.
הציעו מימוש למבנה הנתונים כך שהעלות לשיעורין של כל פעולה על המבנה תהיה מינימלית.
פתרון:
נרצה להשתמש במערך דינמי. ניצור מערך דינמי כך שנחליט שבשלב מסויים אנחנו נמחק את כל מחצית האיברים הגדולים ביותר. כיצד נדע מיהם הגדולים ביותר? באמצעות סלקט. נפעיל סלקט על האיבר ה$\frac{n}{2}$ בגודלו. כעת נתאר מימוש:
כאשר נרצה להוסיף איבר למבנה הנתונים אנחנו נכניסו באינדקס האחרון מהסוף כך שאנחנו יכולים להכניס בו. כלומר אם במבנה כרגע k איברים (התחלנו מאינדקס 1) אזי הוא יוכנס באינדקס $k+1$. (נעיר כי כאשר נגיע לשלב מסויים בו אין מקום להכניס - בדומה למה שראינו בהרצאה נכפיל את גודל המבנה, ע"י יצירת מבנה בגודל $2S$ והעתקת האיברים אליו, בהמשך נסביר מדוע לשיעורין יצא קבוע)
כעת, נרצה לבצע פעולה ב' ולמחוק את הגדולים ביותר. נפעיל פעולת סלקט על האיבר ה$\frac{n}{2}$ בגודלו. כעת אנחנו נעבור על המערך בצורה לינארית ונעתיק את כל האיברים שקטנים מהאיבר החציון בגודלו. הם יועתקו למערך חדש בגודל $\frac{|S|}{2}$.
המבנה די ברור - כעת נותר להסביר מדוע לשיעורין זה יצא קבוע.
ננתח באמצעות שיטת הבנק:
נשים לב שהפעולות הכבדות - שתיים כאלו - העתקת מערך ומחיקת חצי מערך, יהיו כבדות כאשר יבוצעו. כעת נסביר לשיעורין מה יקרה:
לכל איבר ניתן 4 מטבעות:
1. מטבע ראשון להכנסתו
2. מטבע שני להעתקתו למערך חדש במידת הצורך
3. מטבע שלישי לביצוע פעולת סלקט
4. מטבע רביעי למעבר לינארי על איברי המערך והעתקתו למערך חדש
כעת ננתח פעולת הכנסה: נשלם על האיבר בעת ההכנסה מטבע אחד. נותרו לו 3 מטבעות לשימוש עתידי כלומר בהינתן k איברים יש כעת בבנק $3k$ מטבעות. נרצה להכפיל את גודל המערך אם נזדקק לכך - מה שנצטרך יהיה שימוש בk מטבעות להעתקת k איברים. ניצור מערך חדש בגודל $2k$, ונגדירו לכל $1 \leq i \leq k$ כי $B[i] = A[i]$. סה"כ זו לולאת פור פשוטה על k איברים והשתמשנו בk מטבעות.
כעת לפעולת המחיקה - נרצה לבצע סלקט ולכן נגנוב מהבנק k מטבעות כי סלקט על k איברים יעלה $O(k)$, ואז נרצה לבצע מחיקה של חצי מהמערך, נקצה מערך בגודל $\frac{k}{2}$ ונעתיק אליו את האיברים שקטנים מאיבר החציון, מה שיעלה עוד k מטבעות. סה"כ ניצלנו את כל המטבעות בבנק.
ולכן כיוון שהראינו שלכל פעולה חסומה במס' קבוע $4 \leq$, בהכרח העלות לשיעורין של כל פעולה תהיה $O(4) = O(1)$ וסיימנו.
שאלה 3:
נתונה הבעיה הבאה.
קלט: שני מערכים A, B המכילים מספרים ממויינים מהקטן לגדול. כל אחד מהמערכים בגודל n. שימו לב שאין שום הבטחה לגבי יחס הסדר בין איברים מהמערך A לאיברים ממערך B.
פלט: המערכים A, B כך שאיברי המערכים הם מיזוג מערכי הקלט וממויינים מהקטן לגדול , כך שכל איברי A קטנים מאיברי B וכן כל מערך הוא ממויין בפני עצמו.
(כלומר קיבלנו שני מערכים ממויינים, אתה צריך לדאוג ליצור מיון של כל איבריהם כך שהקטנים נמצאים במערך A)
א. הצע אלגוריתם לפתרון הבעיה בזמן $O(n)$ ובמקום $O(n)$
פתרון: ניצור מערך C בגודל $2n$. נעבור על איברי המערכים A, B. בכל שלב נסתכל על האיבר הראשון שעוד לא הכנסנו מהמערך, ונקח את המינימום מבניהם כלומר בהינתן האיבר הראשון נכניס את $min\{A[1], B[1]\}$. נניח והכנסנו את האיבר הראשון בA נקדם את האינדקס להיות 2 ואז בשלב הבא נבדוק את $min\{A[2], B[1]\}$. סה"כ יצירת המערך הזה יעלה $O(n)$ זמן. כעת יש מערך ממויין, נחצה אותו באמצע כלומר נסתכל על האינדקס האמצעי ביותר, כלומר הוא יהיה n כי יש $2n$ איברים, וכעת לכל $1 \leq i \leq n$ נגדיר $A[i] = C[i+n]$ ולכל $n+1 \leq i \leq 2n$ נגדיר $B[i] = C[i]$ וסיימנו. עלה לנו סה"כ n זמן וכן השתמשנו במערך עזר C שלקח מקום של $2n = O(n)$. מש"ל.
ב. הצע אלגוריתם לפתרון הבעיה בזמן $O(nlogn)$ ובמקום $O(1)$. הדרישה היא שהפלט ממש ייכתב על מערך הקלט, כלומר הפעולות ממש מתבצעות במקום. הזכרון הנוסף האפשרי הוא רק $O(1)$
פתרון: נסתכל עם שני פוינטרים. אחד יצביע לסוף המערך הראשון והשני לתחילת המערך השני ונשווה בין הפוינטרים. אם נקבל כי האיבר במערך הראשון קטן מאינדקס במערך השני נחליף בניהם, אחרת אנחנו נדע שהאיבר הגדול ביותר במערך הראשון הוא קטן מהאיבר הכי קטן במערך השני ולכן אין צורך לטפל בו עוד, נקדם את האינדקס של המערך הראשון להיות אחד קודם לפני ואת האינדקס של המערך השני להיות אחד קדימה. כך נשווה

בין $2n$ איברים מה שיעלה $O(n)$ זמן. כעת, קיבלנו שני מערכים כך שבA נמצא הכי קטנים ובB הכי גדולים, נמיין כל אחד ב$O(nlogn)$ וסיימנו.

שאלה 4:

יהי $A[1,..n]$ מערך מספרים שלמים. לכל תת מערך $A[i,...j]$ כאשר $1 \le i \le j \le n$ נגדיר סכום תת מערך $S_{i,j} = \sum_{k=i}^{j} A[k]$. תמיד תת מערך כאן יהיה רציף. להלן בעיית מקסימום תת מערך

קלט: מערך A של מספרים שלמים

פלט: סכום תת מערך מקסימלי של המערך A.

א. הראה אלגוריתם בזמן ריצה $O(n^3)$

ב. הראה אלגוריתם בזמן $O(nlogn)$

ג. הוכח זמן ריצה של האלגוריתם שמצאתם הוא $O(nlogn)$. יש להוכיח ישירות אם אתם מסתמכים על משפט או תוצאה שנלמדו לאורך הקורס יש להוכיח גם אותה.

פתרון:

א. נאיבי מאוד - נעשה שלוש לולאות פור, כאשר הראשונה עוברת מ1 עד i והשנייה מאחד עד j. בכל פעם נבחר זוג אינדקסים (נקח רק מקרים בהם $i \le j$) ונרוץ בלולאה פנימית שתרוץ $k = i$ עד j לחישוב הסכום בפנים. סה"כ במקרה הגרוע הלולאה הפנימית תעלה $O(n)$ ושני החיצוניות ירוצו על n איברים במקרה הגרוע ולכן $O(n^3)$

ב. יודעים כי $T(n) = 2T(\frac{n}{2}) + O(n) = O(nlogn)$. מדוע מחשבה בגישה זו? כיוון שמיון המערך עצמו לא יעזור לי כיוון שאני צריך לשמור על המערך רציף בסדרו המקורי. כעת אם נחליט על הגישה הבאה: נמצא תת סכום רציף מקסימלי בחלק הימני של המערך, נמצא תת סכום מקסימלי רציף בחלק השמאלי של המערך - ונחבר ממש בהפרד ומשול.

המקרה הבעייתי הוא אם המקסימום חוצה משני החלקים - כלומר אם נקבל שהמקסימום מתחיל בצד שמאל של המערך וממשיך לצד ימין.

סה"כ נסתכל על הפסודו הבא

```
maxSubArray(A,left,right)
if lrft==right return A[right]
mid=(left+right)/2
left_max=MaxSubArray(A,left,mid)
right_max=MaxSubArray(A,mid+1,right)
crossmax=maxcrodd(A,left,mid,right)
return max{left_max,right_max,crossmax}
```

כאשר הפונקציה שתמצא את תת המערך האמצעי תעלה $O(n)$ ותחושב כך: נסתכל על האמצע במערך ונתחיל לחפש מקסימום מהשמאל לאמצע, בדומה נחפש מקסימום מימין לאמצע וזה יראה כך:

```
maxCross(A,mid,left,right)
sum=0
for int i=mid to left sum+=A[i], leftsum=max(leftsum,sum)
for int i=mid+1 to left sum+=A[i] rightSum=max(rightsum,sum)
return leftsum+rightsum
```

הסבר: אנחנו למעשה נבצע מעבר לינארי על הסכום מימין ומשמאל לאמצע. אם הוספת האיבר תקטע את המקסימום אנחנו לא נוסיף אותו כי הוא יקטע את המקסימום מהסדרה. אחרת כן נוסיף אותו. סה"כ בסוף נחזיר מקסימום מימין ומשמאל לאמצע וזה הסכום האמצעי.

ג. נוכיח את נוסחת הנסיגה שקיבלנו: נראה כי באופן כללי לi שלבים יתקיים

$$T(n) = 2^i T(\frac{n}{2^i}) + in$$

באינדוקציה על i.

$i = 1$ בדיוק נוסחת הנסיגה המקורית.

נניח נכונות עבור i. נוכיח עבור $i+1$ ונקבל

$$T(n) = 2^i T(\frac{n}{2^i}) + in = 2^i(2T(\frac{n}{2^{i+1}}) + n) + in = 2^{i+1} T(\frac{n}{2^{i+1}}) + n(i+1)$$

כנדרש.

כעת $n = 2^i$ כלומר $i = logn$ ונקבל $T(n) = 2^{logn} * 1 + nlogn = n + nlogn = O(nlogn)$.

נוכיח בדרך שקולה את הטענה באינדוקציה שלמה כי $T(n) = O(nlogn)$.

בסיס - ברור מאליו.

צעד ־ נניח כי לכל $n' < n$ קיים c המקסימלי כך ש$T(n') < cnlogn$. בפרט עבור $T(\frac{n}{2}) < \frac{cnlogn}{2}$. כעת

$$T(n) = 2T(\frac{n}{2}) + O(n) < \frac{cnlogn}{2} + n < \frac{cnlogn}{2} + \frac{cnlogn}{2} < cnlogn = O(nlogn)$$

כעת המעבר באמצע יתקיים כאשר $n < \frac{cnlogn}{2}$ כלומר כאשר $2 < clogn$ כאשר $c > \frac{2}{logn}$ ולכן אם ניקח $c' = max\{c, \frac{2}{logn} + 1\}$ נקבל את הדרוש.

2013 שמואל קליין מועד א

שאלה 3

נתון מערך A בגודל n, כאשר n כפולה של פרמטר k שהוא חזקה של 2 כלומר $n = ky = 2^i y$. ברצוננו לסדר את איברי המערך A מחדש כך ש$\frac{n}{k}$ האיברים הראשונים יהיו הכי קטנים, $\frac{n}{k}$ ההבאים יהיו הקטנים מבין הנותרים וכן הלאה עד שב$\frac{n}{k}$ התאים האחרונים נאחסן את $\frac{n}{k}$ האיברים הגדולים. הראה איך לעשות זאת בזמן $O(nlogk)$

פתרון:

נסיון עם ערימה ייכשל ולא יעבוד. לכן ננקוט בגישה שונה. נבצע סלקט. הרעיון יהיה שנרצה בכל שלב לקבל מצד אחד את הקטנים ביותר ואז להמשיך רקורסיבית עם סלקט. נבצע סלקט פעם ראשונה, על האיבר ה$\frac{n}{k}$ בגודלו. זה יעלה $O(n)$ ולאחריו אנחנו נקבל את כל האיברים הקטנים ממנו מצד שמאל, וכל הגדולים ממנו מימין. כעת נוכל להסתכל על המערך $[\frac{n}{k}+1, ...n]$. יש שם $n - \frac{n}{k}$ איברים. נפעיל עליו סלקט. וכן הלאה..

נראה כי נרצה להפעיל את האלגוריתם c פעמים כפול זמן הפעולה שלו שהיא n. כלומר $O(nc)$. כעת נוכיח $c = logk$.

ההוכחה די פשוטה ־ יש $n = ky$ איברים, בכל שלב מחסרים $\frac{n}{k}$ איברים וכיוון שk כפולה של 2 כלומר $k = 2^i$ כלומר $i = logk$ ואם נפעיל את האלגוריתם $logk$ פעמים נקבל שנעבור על כולם.

הסבר מפורט יותר ־

נחלק את n ב2^i כל פעם עד שנגיע לקבוצה בגודל $\frac{n}{k}$. שנגיע לקבוצה בגודל זה נפסיק כי זה הקבוצה שצריך להגיע לגודלה ־ הכי גדולים שיימצאו שם. סה"כ $\frac{n}{2^i} = \frac{n}{k}$ כלומר $2^i = k$ ולכן $i = logk$, סה"כ נבצע $logk$ ריצות ונקבל $nlogk$.

שאלה 4

נתון מערך $A[1, ...n]$. לא ממויין עם מספרים טבעיים. ברצוננו לבנות עץ חיפוש בינארי שהערכים בקודקודיו הם איברי המערך A.

הוכח שבנייתו תקח $\Omega(nlogn)$

הוכחה: נב"ש שניתן לבנות אותו ב$o(nlogn)$. נבנה אותו ונבצע סריקת $inOrder$ שתעלה $O(n)$, וקיבלנו מיון. בסתירה למיון מבוסס השוואות $\Omega(nlogn)$.

שאלה 5:

מערך A נקרא אונדימודלי אם הוא מורכב מסדרה עולה שאחריו סדרה יורדת, כלומר קיים אינדקס m יחיד כך שהסדרה עולה עד אליו ויורדת לאחריו. כמו כן האיבר $A[m]$ הוא המקסימום היחיד במערך. היחיד ששני שכניו קטנים ממנו.

א. נתון מערך אונדימודלי באורך n. חשב את המקסימום בזמן $O(logn)$ במערך.

פתרון: חיפוש בינארי קלאסי. נבצע חיפוש ובכל שלב נסתכל על הערך האמצעי ביותר. אם מתקיים $A[i] < A[i-1]$ אזי השבירה התרחשה כבר שכן אנחנו במגמת ירידה ולכן נלך לחפש את נקודת השבירה בצד השמאלי. אם $A[i] > A[i-1]$ השבירה עוד לא התרחשה ונלך לחפש אותה בצד הימני. סה"כ $T(n) = T(\frac{n}{2}) + O(1) = O(logn)$.

טענה ־ האיבר המקסימלי במערך אונדימודלי יהיה איבר השבירה הk.

הוכחה: יהי האיבר הk שבו מתרחשת השבירה. בפרט מתקיים לכל $1 \leq i < k$ כי $A[k] > A[i]$ כי הסדרה עולה בקטע זה. בפרט מתקיים לכל $k+1 \leq i \leq n$ כי $A[k] > A[i+k]$ כי הסדרה בשלב זה יורדת. ולכן הוא המקסימלי במערך.

ב. פוליגון הוא קמור אם כל הזוויות הפנימיות שלו קטנות מ180. נייצג פוליגון קמור ע"י מערך של נקודות הקצה שלו הקודקודים $v_1, ...v_n$. כל נקודה נתונה ע"י $v_i = (x_i, y_i)$. נניח שכל הx_i שונים וכל הy_i שונים. v_1 היא הנקודה עם x_1 הקטן ביותר מכל ערכי הx, ויתר הנקודות ממוספרות בניגוד לכיוון השעון .

1. הצע אלגוריתם שמוצא את הערך המקסימלי של x_i בזמן $O(logn)$

א. אנחנו יודעים כי הפוליגון ייוצג במערך, כך שתהיה קיימת נקודה k כך שכל שיעורי הx עד אליה יהיו בעלייה ולאחריה בירידה שכן הפוליגון הוא מרובע וקיים שלב כזה. ולכן זה שקול לבעיה מסעיף א' אותו הפתרון בו נפתור ב$O(logn)$

2. הצע אלגוריתם למציאת y_i המקסימלי ב$O(logn)$

פתרון: הנקודה המקסימלית אמורה להיות היחידה שתקיים $y_{i-1} < y_i$ וכן $y_i > y_{i+1}$, זה בדיוק מתאים למציאת האיבר השבירה במערך האונדימודלי, כאשר נחפש לפי y ושוב ב$O(logn)$.

2013 מועד ב' שמואל

1. בבעיית הוקטור הפרבולי נתון מערך A בגודל n של שלמים שונים וצריך לסדר מחדש את איברי המערך כך שיתקיים:

לכל $1 < i \leq \frac{n}{2}$ (בערך שלם עליון) $A[i] \leq A[i-1]$ כלומר המערך יורד

לכל $n \geq i \geq \frac{n}{2} + 1$ מתקיים $A[i] \leq A[i+1]$ (כלומר המערך עולה.

א. בהינתן וקטור פרבולי תאר אלגוריתם שממיין את הוקטור בזמן $O(n)$

פתרון: מהנתונים בהתחלה המערך יורד ואז הוא מתחיל לעלות. לכן בוודאות הערך המינימלי יהיה נקודת השבירה. ראשית נמצא את נקודת השבירה ע"י הפרד ומשול על האיבר האמצעי ב$O(logn)$. זה יהיה איבר המינימום.

כעת ניצור מערך עזר חדש B ונכניס אליו את האיבר המינימום. כעת אנחנו נוכל לסמן בו אפס למשל. כעת אנחנו יודעים מה נקודת השבירה החדשה, נניח ונקודת השבירה הייתה i אזי האיברים $i+1$ ו$i-1$ הם האיברים הפוטנציאלים הבאים לבדיקה. כיוון שאחד מהם היה הכי קטן שגדול מהמינימום והשני הכי קטן מצד שמאל שגדול מהמינימום. בכל שלב נשווה בין שניהם ונבדוק מי מהם המינימום, אותו נכניס למערך B ונתקדם עם פוינטר לכיוון ממנו מחקנו. סה"כ נתחזק שני פוינטרים לבדיקה ונתקדם. סה"כ $2n$ השוואות וכן $logn$ למציאת האינדקס בהתחלה, ולכן $2n+logn=O(n)$

ב. תנו חסם תחתון (במונחי Ω) לסיבוכיות הזמן הדרושה לפתרון בעיית הוקטור הפרבולי. כלומר, תנו חסם תחתון לזמן ריצת אלגוריתם מבוסס השוואות אשר מקבל מערך כלשהו והופך אותו לפרבולי

פתרון: טענה - נסמן את הזמן הדרוש לביצוע קבלת מערך והפיכתו לוקטור פרבולי בT. אזי $T\in\Omega(nlogn)$

הוכחה: קיבלנו מערך. ביצענו עליו פעולה להפיכתו לוקטור פרבולי שעלתה לנו $O(T)$. נב"ש כי $T\in o(nlogn)$. אזי, נפעיל אלגוריתם מסעיף א' למיון הוקטור הפרבולי וקיבלנו מיון מבוסס השוואות שעלה לנו $o(nlogn)+O(n)=o(nlogn)$, בסתירה לכך שמיון מבוסס השוואות הוא בחסם תחתון של $\Omega(nlogn)$. מש"ל.

3. א. כתוב אלגוריתם שמקבל כקלט מצביע לשורש של עץ בינארי עם n קודקודים ומחזיר תשובה "כן" אם העץ הוא עץ חיפוש ולא אחרת.

פתרון: נעבוד בהפרד ומשול. ראשית נסתכל על הקודקוד ונבדוק, האם מתקיים עבור הקודקוד כי $rootValue>left$ וגם $rootValue<right$. אם כן, נתקדם הלאה רקורסיבית על הבנים $right$ ו$left$ עם תת הבעיה ונבדוק שוב. נשים לב כי עלינו לדעת בכל שלב כי תת העץ הינו כולו קטן מהשורש אם היינו בשמאל או גדול מהשורש אם היינו בימין ולכן לכל מסלול ניצור שני אינדקסים - מינימום ומקסימום, כלומר מקסימום שיכול להיות בו (עד אחד פחות משורש תת העץ הנוכחי) או מינימום במקרה השני. כלומר לכל צומת יהיה טווח מספרים בה מותר להיות. כך נבדוק רקורסיבית עבור כל התאים ויעלה לנו זמן לינארי של $O(n)$.

ב. כתוב אלגוריתם שמקבל כקלט מצביע לשורש של עץ בינארי עם n קודקודים ומחזיר תשובה כן אם העץ הוא עץ AVL ושקר אחרת.

פתרון:

נכתוב אלגוריתם לינארי רקורסיבי.

נרצה לבדוק בכל ביקור בצומת שלנו האם $|H_L-H_R|\le 1$. כעת ניגש רקורסיבית , נבדוק בכל שלב שצד ימין וצד שמאל הם עץ AVL ונתקדם רקורסיבית עד שנגיע לעלים. בעלים נחשב $h=max\{leftH,rightH\}+1$ ומקדם האיזון יהיה $k=leftH-rightH$. סה"כ אם נקבל כי $|k|>1$ אנחנו נחזיר שלא יתכן שזה עץ avl כלומר שקר. אחרת, רקורסיבית נפתור את הבעיה על העץ משמאל ומימין. סה"כ ביקור בכל העץ שיעלה $O(n)$.

4. נתונות שני ערימות המיוצגות ע"י מערכים A,B בגדלים n_1,n_2 בהתאמה. נתון כי כל איברי A גדולים מכל איברי B.

א. הוכח הפרך - בהנחה $n_1>n_2$ האם שרשור שני המערכים כאשר A לפני B מהווה ערימת מקסימום? כלומר המערך $[a_1,........,a_{n_1},b_1,....,b_{n_2}]$ מהווה ערימת מקסימום?

פתרון: מהנתון בהכרח כל איברי A גדולים מכל איברי B, ולכן לכאורה הנתון הראשון של ערימה אמור להסתדר לנו כך ש: האיברים בערימה A הינם כבר מסודרים כערימה, וכן האיברים בB ואם כל האיברים בA גדולים מכל האיברים בB בהכרח גם המינימום של A גדול מהמקסימום של B ולכן התנאי של "הבנים קטנים מהשורש" מתקיים בערימה. כמו כן מתקיים כי האיברים יכנסו ברמה התחתונה מלמטה אם יש מקום ואם לא ברמה חדשה ולכן נקבל ערימת מקסימום חדשה. סה"כ הטענה נכונה.

ב. הוכח הפרך - כמו קודם רק $n_2>n_1$.

פתרון: כאן נקבל הפרכה. נגדיר $A=[10]$ ו$B=[5,1,2]$. שניהם ערימות חוקיות. אם נשרשר נקבל $[10,5,1,2]$ נקבל ערימה עם 10 בראש, שני בנים $5,1$ ואז ברמה השלישית 2 שהוא גדול מ1, בסתירה.

5. הצע מבני נתונים עבור n איברים בשטח $O(n)$ כך שניתן להוסיף לחפש ולמחוק איבר בממוצע ב$O(1)$ וב$O(logn)$ במקרה הגרוע.

פתרון: ניצור טבלת האש עם מצביעים לתוך עץ AVL. נבנה עץ AVL עם מצביעים דו כיווניים לתוך טבלת האש.

סה"כ תפסנו $2n=O(n)$ מקום.

חיפוש - בממוצע נחפש בטבלה ב$O(1)$, ונגיע גם בחיפוש הזה לעץ הAVL כי הפוינטר יעזור לנו.

מחיקה - בממוצע נוכל למחוק ב$O(1)$ בטבלה ובעץ $O(logn)$ במקרה הגרוע.

הכנסה - בממוצע נוכל להכניס ב$O(1)$ ובעץ $O(logn)$ במקרה הגרוע.

אלגו' 1 2024 מועד א שאלה 5

הגדרה: העמדה גלובלית בין 2 מחרוזות מעל א"ב נתון $\sum=\{A,C,G,T\}$ מוגדרת כהעמדה של כל תו במחרוזת אחת מול תו במחרוזת השנייה, או סימן "$-$" שמייצג הכנסה של תו למחרוזת אחת ללא התאמתו לתוך תו במחרוזת השנייה.

הציון של העמדה גלובלית מתואר באופן הבא - כל התאמה בין תווים בהעמדה מעלה את הציון ב1, כל אי התאמה בין תווים מורידה את הציון ב1 וכל העמדה של תו מול "$-$" מורידה ציון ב1.

דוגמה למשל להעמדה גלובלית עם ציון: קו אנכי הוא התאמה, נקודה זה אי התאמה. וכן קו ריק יוריד נקודה גם כן:

$$\begin{array}{cccccccccccccc}
A & T & C & G & A & A & C & T & G & G & C & C & - & - \\
. & . & | & | & . & | & | & | & & & | & | & & \\
T & A & C & G & C & A & C & T & - & - & C & C & A & A
\end{array}$$

ציון ההתאמה עבור הדוגמה הנוכחית הוא אפס ־ 7 התאמות, 3 אי התאמות ו4 העמדות מול "$-$".

א. תארו נוסחה רקורסיבית שבהינתן מחרוזות $S = s_1, ..., s_n$ ו$T = t_1, ..., t_m$ מחשבת את ציון העמדה גלובלית הגדול ביותר בין T לS.

ב. השתמשו בא' ותארו אלגוריתם תכנון דינמי יעיל ככל הניתן שבהינתן 2 מחרוזות T, S מחשב ציון העמדה מקסימלי עבורן.

פתרון:

נראה כי בהינתן שקיבלתי 2 מחרוזות אני חייב לבצע אחד משניים הבאים: או לשים שם – או להשוות ביניהם. אין לי דרך אחרת, אם שמתי – ובמחרוזת השניה היה תו כלשהו, הפסדתי נקודה. אם במחרוזת השניה היה גם – לא יקרה לי כלום. כעת נתאר את הנוסחה הבאה $f(i,j)$ תתאר את סכום העמדה הגלובלית כאשר אני מסתכל על המספרים $1, ...i$ במחרוזת S ו$1, ..j$ במחרוזת T. כמובן $1 \le i \le n$ וכן $1 \le j \le m$. כעת נגדירה כך

$$f(i,j) := \left\{ \begin{array}{cc} 0 & i = j = 0 \\ -j & i = 0 \wedge j > 0 \\ -i & j = 0 \wedge i > 0 \\ max\{f(i-1, j-1) - 1, f(i-1, j) - 1, f(i, j-1) - 1\} & T[i] \neq S[j] \\ max\{1 + f(i-1, j-1), f(i-1, j) - 1, f(i, j-1) - 1\} & T[i] = S[j] \end{array} \right\}$$

לתכנון דינמי נבנה מטריצה A $n \times m$ ונמלא אותה לפי תנאי הבסיס כאן קודם כל, ולאחר מכן נראה כי בהינתן שאני בתא מסויים אני ארצה תמיד או את זה שמעליי למעלה בשמאל או מעליי או בימיני ולכן נמלא בעמודות מלמעלה למטה. הפתרון יהיה בתא $A[n,m]$. סה"כ למילויה יעלה לנו זמן של $O(nm)$ ולמקום יעלה $O(nm)$ אך אם נשמור בכל שלב רק 2 עמודות אחרונות, נוכל להוריד מקום ל$O(n)$.

2012 מועד א' אלגו 1 שאלה 5

יהי $\sum$ א"ב כלשהו ותהי w פונקציית משקל על $\sum$ כך ש$w : \sum \to \mathbb{R}^+$.
משקל של סדרה $s = s_1, ..., s_n$ הוא $\sum_{i=1}^{n} w(s_i)$.
קלט: שתי סדרות x כך ש $|x| = n$ וכן y כך ש$|y| = m$
פלט: משקל סדרה z מקסימלית כך ש$z \subseteq x$ וכן $z \subseteq y$. (תת סדרה ולא חייבת להיות רציפה!)

פתרון:

נגדיר את הפונקציה $f(i,j)$ כאשר $1 \le i \le n$ וכן $1 \le j \le m$ כך ש$f(i,j)$ מחזירה את משקל הסדרה המקסימלית z המשותפת שהיא תת סדרה הן של x עבור איברים $1, ...i$ והן של y עבור איברים $1, ...j$.
בכל שלב נתון: אם מצאנו איבר משותף לשני הסדרות, אזי בוודאות נקח אותו (המספרים חיוביים אנחנו ב$\mathbb{R}^+$!).
אחרת, נבחר האם ברצוננו להסתכל על תת הסדרה $1, ...i-1$ או $1, ...j-1$. כמו כן, אם סדרה כלשהי היא באורך אפס והשנייה לא, אזי תת הסדרה המשותפת שלהן היא בהכרח בגודל אפס כי באופן ריק אין איברים. וכמובן אם אורך שתי הסדרות הוא אפס. כעת נתאר זאת פורמלית:

$$f(i,j) = \left\{ \begin{array}{cc} 0 & i = 0 \vee j = 0 \\ w(x_i) + f(i-1, j-1) & x[i] = y[j] \\ max\{f(i, j-1), f(i-1, j)\} & o.w \end{array} \right\}$$

כעת ניצור מטריצה בגודל $n \times m$. נמלא כל תא ב$O(1)$ תחילה לפי תנאי בסיס וכן אח"כ נראה כי נזדקק בכל שלב או לאחד מעליי או משמאלי או משמאלי למעלה באלכסון ולכן נמלא את המטריצה בעמודות משמאל לימין. הפתרון יהיה בתא (n,m). סיבוכיות המקום תהיה $O(nm)$ ואם נשמור בכל שלב רק 2 עמודות קודמות נוכל לעשות זאת בזמן $O(n)$ נשים לב כי נוכל גם למלא בשורות ואז לשמור מקום של $O(m)$ ולכן סה"כ נתעדף את הבחירה לפי $O(min\{n,m\})$, ובזמן יעלה $O(nm)$.

2012 מועד ב' אלגו שאלה 5

תהי $s = s_1....s_n$ סדרה שהיא מחרוזת. תת סדרה חסומה־k היא תת סדרה $s_{j_1}, ...s_{j_m}$כך שכל שני איברים עוקבים בתת הסדרה הם במרחק לכל היותר k בסדרה s, כלומר לכל j מתקיים $j_{i+1} - j_i \le k$.
קלט: שתי סדרות x, y מאורכים $|x| = |y| = n$ ומס' טבעי k.
פלט: אורך תת סדרה ארוכה ביותר z שנמצאת הן בx והן בy (תת סדרה כמובן) שהיא סדרה חסומה k בx והן בy.

א. כתוב נוסחת נסיגה לפתרון הבעיה.
ב. נתח סיבוכיות זמן ריצה ומקום של הפתרון.
פתרון:
הבעיה דומה למקודם לכן. נגדיר את הפונקציה $f(i,j,dx,dy)$ כאשר $1 \le i \le n$ וכן $1 \le j \le n$ כך שהפונקציה מחזירה את אורך הסדרה המקסימלית z המשותפת שהיא תת סדרה הן של x עבור איברים $1,...i$ והן של y עבור איברים $1,...j$ שהיא גם תת סדרה חסומה k. וכן dx,dy הם ייצגו את המרחק מהמיקום הקודם שהייתי בו. כלומר $0 \le dx \le k, 0 \le dy \le k$. כעת ננסח את נוסחת הנסיגה הבאה

$$f(i,j,dx,dy) = \left\{ \begin{array}{cc} 0 & i=0 \vee j=0 \\ 1+f(i-1,j-1,0,0) & x[i]=y[j] \wedge dx \le k, dy \le k \\ max\{f(i,j-1,dx,dy+1), f(i-1,j,dx+1,dy)\} & o.w \end{array} \right\}$$

הסבר: המרחקים יהפכו לאפס ואפס כי כעת זו נקודת ייחוס חדשה שבה אנחנו נתחיל ממנה למדוד מרחק, אחרת אם אין תו שווה אנחנו בוחרים מקסימום מבין להתקדם כאן או שכאן.
ניצור מטריצה $n \times n \times k \times k$ ונמלא אותה לפי הטבלה. כעת יעלה זמן $O(n^2k^2)$ ונראה כי אנחנו נוכל לחסוך מקום אם נשמור בעמודות הn בכל פעם שניים אחרונות ואז סיבוכיות המקום תרד ל$O(nk^2)$.
שאלה 3 תל אביב 2020 מועד ב' סמס ב'
נתונים שני עצי AVL. כל אחד עם n מס' כלשהם. רוצים להחליף בין $\lfloor \frac{n}{2} \rfloor$ המספרים הכי קטנים של שני העצים. לאחר ההחלפה כל אחד מהעצים צריך להיות עץ AVL תקין. הצע מימוש אופטימלי.
פתרון: נעבוד בסיבוכיות $O(n)$. נסרוק $inorder$ כל אחד משני העצים ולאחר מכן נקבל אותם ממויינים בתוך מערך. כעת נבצע סלקט על האיבר ה$\frac{n}{2}$ בגודלו בשני המערכים. נשתמש במערכי עזר B_1, B_2 לייצג את העצים החדשים. נעבור לינארית על המערך A_1 לאחר סלקט (נעיר כי סלקט ישנה את המבנה שלו ולכן אנחנו נשמור את הצורה המקורית של המערך בשביל שנשמור על המיון שלו) ונעבור לינארית ונעתיק למערך B_2 את חצי האיברים הקטנים ביותר וכן נעתיק מהעץ A_2 את חצי האיברים הגדולים ביותר ובדומה נעשה עבור המערך השני.
כעת קיבלנו שני מערכים ־
מערך B_2 מכיל בחציו הראשון את חצי האיברים הקטנים של A ובחצי השני את חצי הגדולים של B בסדר ממויין!
מערך B_1מכיל בחציו הראשון את חצי האיברים הקטנים של B בסדר ממויין ובחציו השני את חצי הגדולים של A בסדר ממויין!
כעת נוכל למיין גם כל אחד מהמערכים B_1, B_2. נעשה מערכי עזר נוספים C_1, C_2
אסביר כעת על B_1 כיצד נמירו ועבור B_2בדומה.
נרצה למיין את B_1. נסתכל עם שני פוינטרים, אחד לאיבר הראשון והשני לאיבר האמצעי בגודלו (ועוד אחד) שממנו מתחילים אלו שהכי גדולים. כעת נשווה כל פעם זוג איברים ונבחר את המינימלי לשים באינדקס שבו אני נמצא במערך C_1 ונקדם תמיד את האינדקס בC_1. וכן נקדם את האינדקס של החצי במערך B_1 שלקחנו ממנו איבר. ככה נקבל מיון של המערך.
כעת קיבלנו שני מערכים ממויינים C_1, C_2. נבנה מהם רקורסיבית עץ AVL ע"י בחירת האיבר האמצעי ביותר וככה ברקורסיה כפי שלמדנו בתרגול ־ זה יעלה $O(n)$
סה"כ אם נסכם: $inorder$ יעלה $2n$, סלקט על שניהם עוד $2n$. שמירה של מערכים לפני ביצוע סלקט עוד $2n$ להעתקה. העתקת חצי הגדולים מכל אחד לשני עוד $2n$. מיון שני המערכים ע"י השוואת אינדקסים ופוינטרים עוד $2n$. בניית עץ AVL מכל אחד עוד $2n$. סה"כ $12n = O(n)$ זמן וכן $O(n)$ מקום.
שאלה 5 **תל אביב 2020 מועד ב' סמס ב'**
נועם רוצה לממש תור עדיפויות (תומך בהכנסה הקטנת מפתח ומחיקת מינימום) תחת ההנחה הבאה: מפתחות האיברים בתור הם שלמים ונמצאים כל רקע נתון בטווח $[0, logn]$ כאשר n הוא מס' איברים נוכחי בתור
ירצה להשיג
* אתחול תור מתוך סדרה נתונה של m איברים בטווח $[0, logm]$ בסיבוכיות זמן $O(m)$
* הכנסה, מחיקת מפתח והחזרת מינימום ב$O(loglogn)$
פתרון:
נשתמש בעץ AVL. לכל איבר נשמור שדה ־ האיבר המינימלי ביותר בתת העץ השמאלי שלו וכן בימני. קל לעדכן את השינויים הללו.
הכנסה ־ נמיין את המערך בזמן $O(m)$ כיוון שנתון טווח מסויים ולכן אם נרצה לבצע מיון בסיס פשוט ואז נבנה רקורסיבית ממנו עץ AVL ע"י בחירת חציון בכל שלב וככה על שני חצאים ברקורסיה. סה"כ $O(m)$
הכנסה לעץ ־ בעץ אנחנו יודעים כי מס' המפתחות המקסימלי שיכולים להיות הוא $logn$ כי מס' המפתחות הוא ייחודי ואם האיברים הם מהתחום $[1, logn]$ אזי בוודאות ערך עליון של $logn$ זה מס' האיברים הנוכחי כרגע בעץ. לכן הכנסה לעץ תהיה ב$logk = loglogn$ כאשר k הוא גודל הקלט כמו בעץ AVL רגיל. כיוון שהחזקנו שדה שמחזיר את ערך איבר המינימום נדע שאיבר המינימום בעץ הוא השדה של השורש ונחזירו ב$O(1) = O(loglogn)$ וכן מחיקת מינימום לשם כך נצטרך 1 לחפש 2 למחוק ־ 2 פעולות שעולות $logk$ כאשר k גודל עץ וכפי שאמרנו גודל עץ מקסימלי $logn$ ולכן $2loglogn = O(loglogn)$.

שאלה 6 תל אביב 2020 מועד ב' סמס ב'

נגדיר טיפוס נתונים סדרה דינאמית:

א. $init(A)$ אתחול מבנה מתוך מערך נתון A בגודל n. $O(n)$

ב. $Get(i)$ החזרת אינדקס i בסדרה. זמן נדרש $O(logn)$ בתוחלת.

ג. $Replace(x,y)$ שינוי כל המופעים של הערך x בסדרה לערך y. זמן נדרש $O(logn)$ בתוחלת.

הצע רעיון למימוש.

פתרון:

אמרו תוחלת נשתמש בהאש. כמו כן נשתמש בעצים הפוכים לפי $union - find$.

טבלת האש עם פוינטרים לאיברים יעלה $O(n)$ לתוך איברים ביוניון פיינד

נגדיר שקבוצה ביוניון פיינד תהיה כל האיברים בסדרה עם אותו ערך. בשורש של הקבוצה ישמר הערך של הקבוצה.

מכל איבר בתוך מערך קלט A יישמר מצביע לצומת המתאים בUF. מכל ערך ששמור בהאש ישמר מצביע לשורש הקבוצה שמתאים לערך זה ביוניון פיינד

אתחול - נעבור על A, כל ערך שנעבור בו לראשונה (נדע באמצעות האש) יוכנס גם לטבלה וגם עם $makeSet$ כצומת חדש ביוניון פיינד, נשמור מצביע הן מהמערך והן מהטבלה לצומת זה. אם נתקל בערך שכבר ראינו נאחד אותו עם שורש קיים שמייצג כבר ערך זה.

$geti$- נגש לתא i בתוך מערך A. נבצע $find$ על האיבר ונחזיר מזהה קבוצה. גובה העץ ההפוך $O(logn)$ וכך סיבוכיותינו [הרי מטפסים מלמטה לשורש]

$replace$- נחפש בטבלת האש x וy. אם y לא קיים זה המקרה הטוב שלנו - פשוט מחליפים y בx(מוציאים ומחליפים מטבלה ומעדכנים שורש של עץ)

אם כן קיים נרצה לבצע $union$ לקבוצה של x ולקבוצה של y וכן נמחק את x מטבלת האש. סה"כ $O(logn)$

שאלה 8 תל אביב 2020 מועד ב' סמס ב'

תארו מבנה נתונים יעיל ככל הניתן אשר מתחזק קבוצה של קטעים סגורים אינטרוולים על הישר הממשי $\mathbb{R}$. תומך ב -

א. $init$ מאתחל מבנה נתונים ריק

ב. $Insert(x,y)$ מקבל שני ערכים ממשיים $x < y$ ומכניס למבנה קטע $[x,y]$. הפעולה מחזירה מצביע לאובייקט שייצג את הקטע שהוכנס.

ג. $delete(p)$- מקבל מצביע לקטע [] שנמצא במבנה ומוחק אותו

ד. $containCount(x)$ - מקבל נקודה $x \in \mathbb{R}$ ומחזיר את כמות הקטעים $[a,b]$ אשר נמצאים כעת במבנה הנתונים ומכילים את הנקודה. כלומר קטעים עבורם $a \leq x \leq b$.

חשוב - ניתן להניח שניתן לשמור מס' ממשיים במילת מחשב.

פתרון:

נממש עם שני עצים. עץ אחד יכיל את ההתחלה של הקטעים ועץ שני את סוף הקטע. שני העצים כמובן AVL.

אתחול כמובן ב$O(1)$.

הכנסה נכניס את x לעץ T_1 שייצג התחלה של קטעים וy לעץ T_2 שייצג סוף קטעים. הכנסה לעץ AVL כמובן $O(logn)$

מחיקה - נקבל מצביע לקטע, כעת נחפש את x של הקטע וy של הקטע בשני העצים ונמחק אותם. כמובן $O(logn)$

הפעולה האחרונה - מה שנעשה יהיה להוסיף שדה בו לכל איבר יהיה דרגתו . אנחנו נרצה להחזיר מס' הקטעים שקטנים שמתחילים ממה שקטן מאיקס, פחות מס' הקטעים שנגמרים במשהו שקטן מאיקס. סה"כ יעלה הוספת הדרגה $O(logn)$.

שמואל קליין 2014 מועד א'

שאלה 1

השאלה: נתון עץ בינארי חופשי שאיננו מושרש כך שלא קיים בו צומת שדרגתו 2. הוכח שאם יש לעץ $k > 1$ עלים אזי יש לו $2k-2$ צמתים.

הוכחה: נוכיח באינדוקציה את הטענה.

בסיס: $k = 2$, אזי בעץ יש 2 עלים , נראה כי או ששניהם בנים של איבר יחיד ואז זה לא יתכן כי יהיה לו דרגה 2 בסתירה, או שהם עלים בודדים בעץ וזה המצב ומדובר בעץ עם שני קודקודים בודדים כלומר יש בו 2 צמתים ואכן $2*2-2=2$.

צעד: נניח שהטענה נכונה לכל עץ עם $k' < k$ עלים. יהי עץ בינארי חופשי עם k עלים. נסתכל על קודקוד שרירותי. או שבחרנו עלה, ונבחר קודקוד אחר.

או שהקודקוד בעל דרגה 3 לפחות, כיוון שדרגה 2 לא תתכן.

כעת לקודקוד יש שלושה תתי עצים. נסמן את תתי העצים A,B,C כאשר הם מחוברים לקודקוד יחיד נסמנו x.

כל אחד מהם מקיים את הנחת האינדוקציה. נסתכל על מס' הקודקודים בתת העץ A למשל יחד עם איקס וכיוון שהעץ לא מושרש נסתכל עליו כעלה. לכן יש סה"כ k_1+1 עלים. ולכן בתת העץ יש $2(k_1+1)-2=2k_1$ קודקודים, ובדומה בשני העצים האחרים $2k_2, 2k_3$. נראה כי ספרנו את קודקוד הראש כמה פעמים יותר מדי - פעמיים, ולכן נחסר שניים ונקבל

$$2(k_1+k_2+k_3)-2=2k-2$$

שכן מס' העלים בעץ הגדול זה סכום העלים של העצים הקטנים. מש"ל.

שאלה 2

השאלה: תאר כיצד ניתן למיין n מספרים שלמים שכולם מהטווח $[0,n^2-1]$ בסיבוכיות אופטימלית.

פתרון: נשתמש במיון בסיס. נגדיר את הבסיס להיות n ולכן $d=log_n(n^2-1)\le log_n(n^2)=2$, ונקבל סיבוכיות מיון של

$$O(d(n+R))\le O(2(n+n))=O(4n)=O(n)$$

לא ניתן בפחות מכך, חייבים לעבור על כל האיברים לפחות פעם אחת.

שאלה 5:

ברצוננו להשתמש במבנה נתונים סטטי (הרשומות נתונות מראש ואין בהן שינוי) כאשר המפתחות מס' טבעיים. נרצה לענות על השאילתה מהסוג הבא: בהינתן מס' j מצא את המס' k הקרוב ביותר לj שיופיע במבנה הנתונים. בהנחה שיש מקום בזכרון למבנה נתונים שלנו, באיזה מבנה נתונים תשתמש? תאר אלגוריתם שעונה על הדרוש וחשב סיבוכיות זמן.

פתרון:

נשתמש בעץ AVL. נבנה אותו תוך כדי ההכנסה וכן הכנסה ומחיקה וחיפוש יעלו $O(logn)$. אתחולו יעלה $O(1)$. לבנות אותו $O(n)$ אם המספרים ממויינים ו$O(nlogn)$ אחרת.

נוסיף סוויץ' קטן לעץ, שכל תא יכיל את העוקב של האיבר הנוכחי וה$preseccssor$. דבר זה יעלה מס' פעולות קבוע ולא ישפיע **אסימפטוטית** על המבנה.

כיצד נוסיף פעולת עוקב?

קודם כל נחפש צומת j. אם לא קיים נוסיף אותו לעץ, ובסוף נמחוק אותו.

נחלק למקרים:

אם יש לצומת j תת עץ ימני, אזי העוקב יהיה האיבר הקטן ביותר בתת העץ הימני.

אם אין, נטפס למעלה בעץ כל עוד אנחנו בן ימני ונחזיר את ההורה הראשון שנהיה בן שמאלי שלו.

$presecssor$ בדומה. כעת נבצע מינימיזציה (בחירת מינימום) מבין העוקב והקודם, וזה יהיה הכי קרוב.

אם אין קודם, כלומר הוא הכי קטן, נחזיר תמיד עוקב. אם אין עוקב כלומר הוא הכי גדול נחזיר תמיד קודם.

סה"כ זה יכול לעלות $O(logn)$ במקרה הגרוע ולא ישפיע על הכנסה ומחיקה שממילא $O(logn)$

שמואל קליין 2015 מועד א' שאלה 4

נסמן בn_i את מס' הקודקודים בעץ בינארי שיש להם i בנים. כמובן $i\in\{0,1,2\}$. הוכח באינדוקציה על המבנה שלכל עץ בינארי לא ריק מתקיים $n_0=n_2+1$. (אם $n_1=0$ אין קודקודים פנימיים עם בן אחד בלבד, כלומר העץ שלם, אבל המשפט יהיה נכון לכל ערך של n_1). רמז - התייחסו לקודקוד העליון בעל 2 בנים.

הוכחה:

נוכיח באינדוקציה על מס' הקודקודים.

בסיס - $n=1$ אזי יש קודקוד אחד, כלומר אין קודקוד עם 2 בנים $n_2=0$ ואכן $n_0=1=1+0=1$.

צעד: נניח נכונות לכל עץ עם $n'<n$ קודקודים. נסתכל על הקודקוד העליון בעץ בעל 2 בנים. נסמן את בניו x_1,x_2. בהכרח הם מקיימים את הנחת האינדוקציה. כלומר, $n_{0_1}=n_{2_1}+1$ וכן $n_{0_2}=n_{2_2}+1$. כעת,

מס' הקודקודים n_{0_1} הוא מס' הקודקודים בעץ שיש להם אפס בנים. נראה כי כאשר נסתכל על העץ הכולל, $n_{0_1}+n_{0_2}=n_0$ כיוון שנחבר את שני תתי העצים לשורש אחד, ולא נוסיף עלים שיש להם אפס ילדים. ולכן

$$n_0=n_{0_1}+n_{0_2}=n_{2_2}+1+n_{2_1}+1=n_{2_1}+n_{2_2}+2$$

אם כן, נשים לב כי $n_{2_1}+n_{2_2}=n_2-2+1=n_2-1$ כיוון שכעת הוספנו לעץ הגדול קודקוד עם 2 בנים, x_1 וx_2 ולכן מס' הקודקודים שיש להם 2 בנים ירד ב2 אך התווסף אחד - שורש העץ.

סה"כ אכן $n_0=n_2+1$.

מקרה שני - אם לקודקוד העליון אין 2 בנים יש בן יחיד, אזי נסתכל על הבן היחיד שלו x הוא מקיים את הנחת האינדוקציה $n_{0_1}=n_{2_1}+1$, כעת כאשר נצרף את העץ מס' הקודקודים שיש להם בן יחיד יגדל באחד, נראה כי $n_0=n_{0_1}$ (לא השתנה) וגם $n_{2_1}=n_2$ ולכן מה שהשתנה זה רק n_1 שלא קשור לנוסחה ואכן $n_0=n_2+1$.

2018 מועד ב' שמואל שאלה 1

עץ טרינארי הוא עץ בו לכל קודקוד לכל היותר 3 בנים. נגדיר שדה נוסף $C(v)$ לכל קודקוד בן נרצה לאחסן את גודל תת העץ (מס' קודקודים) ששורשו v כולל v עצמו.

בהינתן T טרינארי עם n קודקודים הראה כיצד למצוא $C(v)$ לכל קודקוד v בזמן $O(n)$

פתרון: נפעל רקורסיבית, בדומה לעץ בינארי, נלך עד לעצים ועבור כל קודקוד נגדיר $C(v) = C(v.left)+C(v.right)+C(v.middle)+1$, אם הקודקוד הוא עלה אזי $C(v) = 1$ בלבד (הוא עצמו), אם אין לו בן שמאלי או ימני כמובן נגדיר בהתאם. נתבונן בפסודו =

```
AddC(T)
if T.v.hasRight==null and T.v.hasLeft==null and T.v.hasMiddle=null so T.v.c=1
if T.v.hasRight==null and T.v.hasLeft==null so T.v.c=1 + T.v.middle.c
if T.v.hasRight==null and T.v.hasMiddle==null so T.v.c=1 + T.v.Left.c
if T.v.hasMiddle==null and T.v.hasLeft==null so T.v.c=1 + T.v.right.c
o.w
t.v.c=addc(v.left)+addc(v.right)+addc(v.middle)+1
```

כמובן פסודו בסיסי ביותר להבנת רעיון.

2018 מועד ב' שאלה 2

נתונים m מערכים $A_1,...A_m$. לא ממויינים שאיבריהם כולם מס' שלמים בתחום $[1,k]$. יהי n המס' הכולל של האיברים בכל המערכים יחד. כתוב אלגוריתם שממיין כל המערכים בזמן $O(k+n)$, כלומר לאחר הרצת האלגוריתם כל אחד מהמערכים יכיל אותם איברים שהכיל לפני ההרצה אך כעת בכל מערך יהיו ממויינים

פתרון:

נעתיק את כל איברי המערכים למערך גדול, בגודל n, כאשר נשמור שדה בו יציין עבור כל איבר מאיזה מערך הגיע.

כעת, יש לנו מערך B בגודל n. כל האיברים בו בטווח $[1,k]$. נבחר בסיס k ונקבל מיון של כל האיברים ב־ $d = log_k k = 1$ ולכן זמן מיון $O(1(k+n)) = O(k+n)$. כעת נעבור על המערך B בצורה לינארית, נתקל באיבר שהיה במערך m_1 למשל אז נכניס אותו למערך הזה במיקום הראשון ונקדם את האינדקס בו אנחנו מכניסים איברים למערך זה. כך נכניס את כולם באופן ממויין (כאשר בכל פעם נעבור על האיבר הכי קטן בB ונשלח אותו למערך שהיה בו במיקום הרלוונטי). סה"כ זה יעלה לנו $O(n)$ מעבר לינארי ולכן סה"כ $n+k+n = O(n+k)$ מש"ל.

2020 מועד ב' שאלה 2 שמואל

הגדרה: סדרה $S = s_1,...,s_n$ נקראת $4-8$ עולה אם לכל $1 \le i \le n-1$ מתקיים $4s_i \le s_{i+1} \le 8s_i$. להלן תיאור בעיית תת סדרה $4-8$ עולה ארוכה ביותר:

קלט: סדרת מספרים S

פלט: אורך תת סדרה $4-8$ עולה ארוכה ביותר.

הבעיה ניתנת לפתרון בתכנון דינמי.

פתרון:

הפתרון הנאיבי הוא ללכת על 2^n תתי הסדרות ולבדוק אם התנאי מתקיים מה שיתן $O(n*2^n)$

כעת ניגש לנוסחת הנסיגה ־ נרצה להגדיר פונקציה שתחזיר לנו תת סדרה ארוכה ביותר.

נגדיר $f(i)$ להיות תת הסדרה ה$4-8$ עולה הארוכה ביותר באינדקסים $1,..i$ שמסתיימת באינדקס i. נראה כי ־

$$f(i) := \left\{ \begin{array}{cc} 1 & i=1 \\ max_{1 \le j<i}\{f(j)\}+1 & 4s(j) \le s(i) \le 8s(j) \end{array} \right\}$$

הוכחת נכונות: אם $i=1$ אזי אנחנו מסתכלים על תת הסדרה של האינדקסים $\{1\}$. בפרט היא סדרה $4-8$ עולה באופן ריק ולכן אורכה המקסימלי שכולל אותה הוא אחד.

אחרת, נרצה לעבור על כל האינדקסים שקודמים לi בסדרה הנוכחית כיוון שזו תת סדרה ניתן לבצע זאת בדילוגים כאשר התנאי היחידי יהיה שאם בחרתי $1 \le j < i$ אזי אני רוצה שיתקיים עבור האינדקס הנוכחי שלי i כי $4s(j) \le s(i) \le 8s(j)$ כיוון שj הוא האינדקס הקודם לי בתת הסדרה.

פתרון תכנון דינמי: נבנה מערך בגודל n ונסמנו A. נמלא $A[1] = 1$. כעת נמלא את המטריצה לפי תנאי נוסחת הנסיגה כלומר משמאל לימין כאשר כל תא מחשב מקסימום על התאים הקודמים לו, סה"כ במקרה הגרוע למשל אם $i=n$ אנחנו נעבור על כל $1 \le j < n$ כלומר על $n-1$ איברים ולכן מילוי תא בודד יעלה $O(n)$. סה"כ נמלא n תאים ולכן סיבוכיות הזמן תהיה $O(n^2)$, כמו כן סיבוכיות המקום תהיה $O(n)$ כגודל המערך. נשים לב שלא נוכל לחסוך במקום כיוון שבכל שלב אנחנו זקוקים לכל התאים הקודמים. אם כן, כיוון שזו תת סדרה היא יכולה להסתיים בכל אינדקס. מההגדרה ־ $f(i)$ תחשב לנו תת סדרה מקסימלית שנגמרת באורך זה. ולכן בשביל למצוא את הפתרון הוא יהיה $max_{1 \le i \le n}\{f(i)\}$ כלומר נעבור על כל התאים פעם נוספת למצוא פתרון מקסימלי ־ זה לא יפגע כמובן בסיבוכיות האסימפטוטית כי מעבר כזה הוא לינארי כלומר $O(n)$. מש"ל.

2020 **מועד ב' שאלה 3 שמואל**

מבנה S תומך בארבע פעולות על מערך $A[1,...n]$ כמו כן מערך זה מכיל מס' טבעיים שונים.

א. $init(A)$ בזמן $O(n)$

ב. $find10(i,j)$ מחזיר את האיבר k שהוא העשירי בגודלו בתת המערך בטווח i עד j ואת האינדקס k של האיבר העשירי. במקרה בו $j-i<10$ יחזיר $null$. זמן $O(1)$
ג. $update(i,k)$ ־ חסום בזמן $O(1)$. מעדכן במערך A אינדקס בתא i להיות k.
ד. $read(i)$ חסום בזמן $O(1)$. מחזיר איבר k באינדקס i במערך k.
הוכח הפרך ־ קיים מבנה נתונים שעונה על דרישות הנ"ל.
פתרון:
ננסה להפריך. אמנם פעולות א, ג וד' נראות לגיטימיות וניתן לבצען בקלות על מערך בסיבוכיות הנדרשת. אם כן הפעולה בסעיף ב' נראית אבסורדית. ננסה להוכיח שלא קיים:
נב"ש שקיים מבנה כזה ונרצה להפעיל פעולה $find10(i,j)$.
במידה ויש פחות מעשרה איברים ־ בסדר, נמיינם. אחרת, נניח $n>10$. נפעיל פעולה $find10(1,n)$. קיבלנו כעת את האיבר העשירי בגודלו במערך ב$O(1)$ זמן (נשמע פסיכופתי הרי הוכחנו שסלקט מתבצע ב$O(n)$ אבל ניחא, נמשיך). כעת נוציא אותו ונשים אותו במערך עזר B במיקום עשר (פורמלית אין פעולת הוצאה מהמבנה ולכן נשים בו אנסוף, ונעתיקו למערך אחר). כעת, לאחר הוצאת האיבר a_{10} נשארנו עם האיברים $a_1,....a_9,a_{11},....$. כעת האיבר ה10 בגודלו יהיה a_{11}. גם אותו נוציא ונשים באינדקס 11. כך נמשיך עד שנגיע למצב בו לא ניתן יותר לבצע פעולת $find10$. כלומר, קיבלנו שבמערך יש פחות מ10 איברים. סה"כ הצלחנו למיין את האיברים $a_{10},a_{11},....a_n$ וכעת נותר למיין מס' קבוע של 9 איברים, ב$O(1)$ כמובן, ולכן סה"כ ביצענו סדר גודל של n הוצאות מהמערך (שוב, בערך) ופעולת המיון בסוף שלקחה $O(1)$ וכן n קריאות ל$find10$ ולכן מיינו את המערך ב$O(n)$ זמן, בסתירה לחסם תחתון $\Omega(nlogn)$.
מבחן תל אביב $2019AA$ שאלה 15
נרצה לתחזק משחק סודוקו. המשחק מתנהל על לוח $n\times n$ כאשר מטרת המשחק היא למלא את כל הלוח במספרים מ1 עד n. אסור שיהיה אותו מספר פעמיים באותו עמודה או שורה. השחקן יכול לבצע:
א. $write(i,j,number)$ ־ כותב מס' $number$ במיקום $[i,j]$ בלוח. מהלך חוקי אם הוא לא מפר כלל שורה ועמודה אם לא חוקי יש להודיע למשתמש ניתן להניח שאין איבר במיקום זה בעת ההכנסה
ב. $contains(i,j)$ בודקת האם קיים איבר במיקום i,j בלוח. אם קיים מחזירו למשתמש
ג. $erase(i,j)$ מוחקת איבר במיקום i,j בלוח. ניתן להניח שקיים איבר במיקום זה בעת המחיקה
ד. won מחזירה למשתמש אם הוא ניצל (לא נממש זאת)
סעיף א (7 נקודות) ־ תאר מימוש ל3 הפעולות הראשונות כאשר כל אחת רצה בזמן $O(1)$ בתוחלת וכן זמן ומקום האתחול הוא $O(n^2)$.
פתרון:
אמרו תוחלת נשתמש בהאש. ראשית נבנה את המטריצה שלנו בגודל $n\times n$ והנה אתחלנו את המשחק. מה שנעשה גם יהיה לשמור $2n$ טבלאות האש שבכל אחת n איברים ־ כלומר נשמור לכל שורה טבלת האש משלה, ולכל עמודה טבלת האש משלה. סה"כ זה $2n^2$ סיבוכיות מקום שיעלה הנתון. כמובן שנוכל להגדיר שנשמור את הטבלאות באמצעות מערך עמודות $B[1,....n]$ שכל איבר במערך יהיה מצביע לטבלתה אש שמתאימה לעמודה הi ומערך שורות $C[1,...n]$ בדומה. כעת, אתחלנו את המשחק עם $O(n^2)$ מקום כנדרש. הערה ־כיוון שאפס בוודאות לא יהיה איבר נעדכן תחילה את המטריצה עם אפסים לאורך כולה. כעת,
א. אם נרצה לכתוב ־ נגש לאיבר המסויים. נבדוק בטבלת העמודה הספציפית האם האיבר כבר מופיע בשורה הספציפית הזו, וכנ"ל בטבלת האש של העמודות אם מופיע כבר בעמודה הספציפית הזו. סה"כ חיפוש בטבלאות האש יעלה $2O(1)$. אם הוחזר בשני המקרים שלא, אז נכניס את הערך וכמובן שגם נכניסו בהתאם לטבלאות האש המתאימות (שכן הכנסה לטבלת האש תעלה $O(1)$) לכן סה"כ פעולת כתיבה ב$O(1)$
ב. בדיקה פשוטה ־ ניגש לאיבר במיקום $[i,j]$. אם יש שם ערך שאיננו אפס נחזיר אותו אחרת נודיע למשתמש שאין שם ערך.
ג. בדומה להכנסה ־ כעת נגש ונמחק אותו ונמחק את הערך שלו מטבלאות ההאש המתאימות לשורה ולעמודה שלו, כמובן מחיקה מהאש בזמן קבוע ולכן סה"כ $O(1)$.

סעיף ב (3 נקודות) ־
בכל הסעיפים הבאים סיבוכיות זמן ומקום האתחול הוא $O(1)$ וסיבוכיות המקום בכל רגע נתון הוא $O(k)$ כאשר k מס' איברים במבנה.
סעיף עזר ללא קשר למשחק ־ תאר כיצד ניתן לייצג וקטור באורך m שתומך בפעולות הכנסה למקום הi, קריאה מהמקום הi ומחיקה מהמקום הi בזמן $O(logm)$ במקרה הגרוע לכל פעולה.
פתרון: נייצג את הוקטור באמצעות עץ AVL. בוקטור m איברים לכן נאחסנו בתוך עץ AVL וכעת נוכל להכניס לחפש ולמחוק ב$O(logm)$. הערה ־ כמובן שהמפתח יהיה האינדקס וה$value$ יהיה הערך המספרי.
סעיף ג (5 נקודות) ־ סעיף עזר נוסף ־ תאר כיצד ניתן לתאר מטריצה בגודל $m\times m$ שתומך בפעולות הכנסה מחיקה וקריאה מהמיקום ה(i,j) בזמן $O(logm)$ במקרה הגרוע.
פתרון: שוב עץ AVL. ניצור עץ לפי מיקומי הi ובו לכל ערך יהיה עץ פנימי בגודל m שבו יאוחסנו ערכי הj שמתאימים לערך הi. סה"כ מדובר על מבנה חיצוני בו m איברים ובכל אחד מהאיברים m איברים פנימיים בעץ. חיפוש ־ נחפש אינדקס i בעץ חיצוני ב$O(logm)$ ואז נחפש בתוך הפנימי שוב ב$O(logm)$, מחיקה ־ נגש לערך הi המתאים ושם נבצע מחיקה של הj. אם מחקנו את האחרון נמחק גם את הi החיצוני. סה"כ $O(logm)$. הכנסה ־ אם עוד לא

קיים נייצרו בעץ המקורי ונכניס לו איבר אחד לעץ הפנימי, אם קיים ערך i נחפשו ובתוכו בפנים נוסיף ערך j סה"כ $O(logm)$.

סעיף ד (10 נקודות) - תאר מימוש לשלושת הפעולות הראשונות כך שכל הפעולות רצות בזמן $O(logn)$ וסיבוכיות המקום של המבנה הוא $O(n)$

פתרון: נשתמש בסעיף ג' ונייצג את המטריצה המדוברת באמצעות עץ AVL.

בנוסף נחזיק מערך עמודות, מערך שורות כך שאם איבר מופיע בשורה או עמודה ספציפית יופיע שם אחד. 0 אחרת.

כעת, אם נרצה לכתוב נלך לחפש את המיקום הרלוונטי ונלך לבדוק את מערכי השורות והעמודות, אם יש שם אפס נוכל לשים אחרת לא. הכלה זה פשוט חיפוש האם קיים בעץ, ומחיקה טריוואלי מאוד לעץ העצים - מחיקה מהפנימי, אם בפנימי לא נותרו איברים נמחוק גם חיצוני. סה"כ $O(logn)$ לכל פעולה.

שאלה 16 תל אביב $2019AA$

תאר מימוש של מבנה נתונים עם הפעולות הבאות (x הוא מפתח)

א. $insert(x)$ בזמן אמורטייזד $O(logn)$

ב. $search(x)$ בזמן אמורטייזד $O(logn)$

ג. $delete$ בזמן שיקבע בהמשך

א. ממש $delete$ בזמן אמורטייזד $O(logn)$

פתרון: עץ AVL, אין צורך אפילו אומרטייזד אלא חיפוש מחיקה והכנסה זה תמיד ב$O(logn)$ במקרה הגרוע. ובפרט בממוצע.

ב. ממש $delete$ בזמן אמורטייזד $O(1)$

פתרון:

נראה כי ביצוע פעולה כזו לא יקרה יותר מדי באמת בלוגאן. ננתח לפי שיטת הבנק ובעת הכנסה אנחנו נקנה לאיבר $2logn$ מטבעות. $logn$ להכנסה כרגע ו$logn$ לשימוש עתידי בעת המחיקה. כאשר נרצה למחוק נשתמש בהם ולכן לא שילמנו כלום, עבור הכנסה יעלה $2logn = O(logn)$ ולהוצאה $O(1)$.

תל אביב $2019BA$ **שאלה 4**

נתונה ערימת מינימום בינארית בגודל n. תארו אלגוריתם שמפצל אותה ל3 ערימות בינאריות בראשונה $\frac{n}{3}$ הכי קטנים, בשנייה האמצעיים ובאחרונה ה$\frac{n}{3}$ הכי גדולים. הניחו $n\%3 = 0$.

פתרון:

הערימה מיוצגת במערך ואם לא נסרוק אותה ונייצגה במערך, נבצע סלקט על האיבר ה$\frac{n}{3}$ בגודלו ונעתיק את האיברים למערך עזר בו ניצור ערימה מ$\frac{n}{3}$ איברים וזה יעלה $O(n)$. בדומה נקח את המערך המקורי ללא האיברים שהוצאנו (למשל נסמן שם אנסוף או נעתיק למערך חדש בגודל $\frac{2n}{3}$) וכעת נמצא את האיבר ה$\frac{n}{2}$ בגודלו שכן לאחר הוצאת $\frac{n}{3}$ הראשונים נשארנו עם $\frac{2n}{3}$ איברים ובשביל למצוא את ה$\frac{2n}{3}$ בגודלו בכל המערך הכולל נצטרך כעת לבצע על ה$\frac{n}{2}$, בכל מקרה נבצע עליו סלקט ונעתיק את כל האיברים שקטנים שווים לו למערך עזר חדש וממנו גם נבנה ערימה בזמן לינארי ולבסוף את האיברים שנשארו הכי גדולים נקח למערך עזר נוסף וממנו נבנה ערימה בזמן לינארי גם כן.

תל אביב $2019BA$ **שאלה 6**

הניחו שבאלגוריתם סלקט נחלק את המערך לרביעיות במקום חמישיות. רשמו נוסחת נסיגה מתאימה לה וכתבו מה סיבוכיות זמן הריצה שלה.

פתרון:

נרצה למצוא את החציון ולאחר מכן את חציון החציונים. נשים לב כי כעת המס' איננו אי זוגי ולכן נגדיר שהחציון של קבוצה בגודל 4 יהיה באינדקס 3 למשל. כעת נמצא חציון חציונים, במחצית מהקבוצות נוכל להפטר מ$\frac{3n}{4}$ איברים ולכן סה"כ נפטר מ$\frac{3n}{8}$ איברים ונוסחת הנסיגה תהיה $T(n) = T(\frac{n}{4}) + T(\frac{5n}{8}) + n = O(n)$.

אם נחליט שחציון של קבוצה כזו יהיה באינדקס 2 נקבל שנפטר בכל שלב מ$\frac{2n}{8}$ ולכן נקבל $T(n) = T(\frac{n}{4}) + T(\frac{3n}{4}) + n = O(nlogn)$

תל אביב $2019BA$ **שאלה 9**

נתונה סדרת איברים $a_1, ..., a_n$ שונים זה מזה. בכל סעיף קבע האם ניתן לבנות מהסדרה עץ AVL בזמן $O(n)$

א. לכל i מתקיים a_i הוא חציון איברי הסדרה עד אליו. נניח שבמקרה וכמות האיברים זוגית אזי החציון יהיה החציון התחתון למשל החציון של קבוצה בגודל 4 יהיה ה2 בגודלו.

פתרון: ניתן לבנות עץ AVL. נעדיף להראות כיצד ניתן למיין מערך כזה ואז לבנות ממנו עץ AVL כמו שלמדנו ב$O(n)$ רקורסיבית.

כיצד נמיין את המערך הזה? בהכרח a_n הוא איבר החציון של המערך. כעת נעבור לa_{n-1} שהוא החציון של מי שנשארו, אם הוא גדול מa_n נשים אותו מימינו ואם קטן משמאלו לפי יחס הסדר בניהם וכן הלאה נמשיך עם האיברים עד שנגיע לאיבר הa_1. סה"כ מעבר לינארי על n איברים ולכן $O(n)$

ב. לכל $1 < i \leq n$ אם i זוגי אזי $a_i > a_{i-1}$ אחרת $a_i < a_{i-1}$.

פתרון: נשים לב שהמערך מקיים $a_4 > a_3 < a_2 > a_1$ וכן הלאה כלומר יש כאן התחלפויות. לא ניתן. נניח בשלילה שניתן ונדפיס $inorder$ ונקבל מיון שבזמן $O(n)$ אך המערך הזה לא ממויין בכלל ולא ניתן למיין אותו בסתירה לחסם תחתון.

תל אביב $2019BB$ **שאלה 5**

נתונה סדרה עם n מספרים וידוע שיש רק $\lfloor log^7 n \rfloor$ מספרים שונים זה מזה. הוכח הפרך - ניתן למיין ב$O(n)$ בתוחלת

פתרון: נבנה טבלת האש בגודל $\lfloor log^7 n \rfloor$, נעבור באופן לינארי על המערך ובכל פעם שנראה איבר חדש נכניסו לטבלה, אם כבר הוא בפנים אנחנו נגדיל את קאונטר המופעים שלו באחד. סה"כ סכום הקאונטרים כמובן יהיה שווה n. כעת נרצה להעביר את האיברים הללו למערך ולבצע מיון על קלט בגודל זה ולקבל

$$O(\lfloor log^7 n \rfloor log(\lfloor log^7 n \rfloor)) = O(n)$$

סה"כ מעבר לינארי על קלט וכן n הוצאות הכנסות וכדומה שבהאש עולה $O(1)$ ונקבל $O(n)$ למיון.

תל אביב $2019BB$ **שאלה 7**

בשאלה זו מתייחסים לקבוצת השלמים הבאה $B = \{1, 2, 4, 8, ..., 2^n\}$.

הציעו מבנה נתונים שתומך בפעולות הבאות. הניחו כי פעולות אריתמטיות רצות בזמן קבוע.

א. $Build(L)$ אתחול המבנה מתוך רשימה L עם m איברים שמפתחותיהם מתוך הקבוצה B. סיבוכיות $O(m+n)$

ב. $Delete(k)$ מחיקת איבר בעל מפתח $k \in B$ זמן $O(1)$

ג. $Num(k)$ החזרת מס' הפעמים בהם המפתח k מופיע במבנה. $O(1)$

ד. $Max()$ החזרת מס' מקסימלי מבין המפתחות במבנה. זמן - הנמוך ביותר שתצליחו.

פתרון:

נראה כי ישנם בהינתן תחום האיברים $[1, ...2^n]$ רק $logn$ מפתחות אפשריים.

ניצור מערך מונים וכן רשימה מקושרת דו כיוונית ממויינת עם כל המפתחות שהמונה שלהם חיובי. כלומר מופיע לפחות פעם אחת בקלט ונתחזק מצביע לסוף הרשימה, במערך המונים יהיה מצביע לאיבר המתאים ברשימה המקושרת. מחיקה תהיה בזמן קבוע למקום המתאים במערך המונים כלומר הקטנת המונה באחד, אם הוא התאפס ניגש באמצעות מצביע קבוע לאיבר ברשימה המקושרת ונמחק משם איבר ב$O(1)$ זמן. Num יתבצע ע"י החזרת המונה המתאים וכן Max יהיה החזרת המפתח שבראש הרשימה המקושרת (הכי גדול).

תל אביב $2019BB$ **שאלה 8**

נתונה מטריצה של מס' שלמים עם m שורות וn עמודות. כל שורה ממוינת בה בסדר עולה. מעוניינים באלגוריתם להדפסת כל איברי המטריצה בסדר ממויין (עולה) בזמן $O(mnlogm)$ וב$O(m)$ זכרון.

פתרון:

אורך של שורה הוא n וכן אנחנו יודעים שהן ממוינות בפני עצמן. ניצור ערימה בגודל m, נכניס אליה את m האיברים הראשונים במטריצה משמאל (הכי קטנים בכל שורה), בוודאות המינימלי יהיה אחד מהם כי הם הקטנים ביותר בכל שורה. נדפיס את המינימלי ונוציא אותו - יעלה $O(logm)$. כעת, נוסיף לערימה את האיבר הבא מאותה שורה שהוצאנו והפוינטר יהיה עליו וכך נמשיך הלאה, כאשר בסה"כ נכניס לערימה mn איברים ונבצע הכנסות והוצאות ולכן סיבוכיות כוללת של $2mnlogm = O(mnlogm)$.

טכניון 2015-2014 שאלה 3

ליואב וליעל יש חבילה עם n קלפים כשעל כל קלף מודפס מספר חיובי. הם מסדרים את הקלפים בשורה ומשחקים משחק שמתקדם בתורות לסרוגין. כל אחד בתורו בוחר האם לקחת קלף אחד או שניים מצדה השמאלי של שורת הקלפים. המשחק מסתיים כאשר כל הקלפים נלקחו והמנצח הוא זה שמחזיק קלפים שסכומם גדול יותר. יעל משחקת ראשונה. בהינתן n קלפים מסודרים בשורה, הציעו ליעל דרך יעילה לקבוע כמה קלפים היא צריכה לקחת בתור הראשון, כך שהיא תבטיח לעצמה את הניצחון (ללא תלות במהלכים שיואב יבחר לבצע), או קבעו שיעל לא יכולה להבטיח לעצמה את הניצחון. הניחו כי הקלט לבעיה נתון במערך a שכך a[0] הוא הערך של הקלף השמאלי ביותר בשורה.

פתרון: נגש לשאלה באמצעות תכנון דינמי. נגדיר פונקציה $f(i,j)$ שתהיה הסכום המקסימלי שניתן להשיג כאשר לוקחים את הקלפים $1 \leq i \leq j \leq n$. נסמן את יעל כשחקן מס' 1 ואת יואב כשחקן מס' 2. נראה כי למעשה הסכום שיעל תרוויח יהיה סכום הקלפים הכולל, פחות הסכום שיואב ירוויח. ולכן בשביל שיעל תוכל למקסם את הרווח שלה היא צריכה לדאוג שיואב ימזער את הרווח שלו. כמו כן בכל שלב יש לשחקן שני אפשרויות: או לקחת קלף אחד מהצד השמאלי כלומר להסתכל כעת על תת הבעיה $f(i+1,j)$ או להסתכל על תת הבעיה $f(i+2,j)$. כעת נסמן את הפונקציות הבאות:

$$f_1(i,j) = \sum_{k=i}^{j} A[k] - min\{f_2(i+1,j), f_2(i+2,j)\}$$

$$f_2(i,j) = \sum_{k=i}^{j} A[k] - min\{f_1(i+1,j), f_1(i+2,j)\}$$

נעיר כי עבור מקרה בסיס $i=j$ נחזיר $A[i]$.
כעת, נראה כי זה יהיה מאוד מסובך לגשת לשאלה בתכנון דינמי. הסיגמה מאוד יקרה. לכן מה שנעשה יהיה יהיה ליצור מערך סכומים, נסמנו S וגם אותו נמלא בתכנון דינמי! כלומר נגדיר תנאי בסיס $S[1]=A[1]$ ולכל $i>1$ נגדיר $S[i]=S[i-1]+A[i]$. כעת מילוי מערך S שכזה יעלה $O(n)$. כעת אנחנו נוכל לשנות את נוסחאות הנסיגה ל

$$f_1(i,j)=S[j]-S[i-1]-min\{f_2(i+1,j),f_2(i+2,j)\}$$

$$f_2(i,j)=S[j]-S[i-1]-min\{f_1(i+1,j),f_1(i+2,j)\}$$

(נשים לב $i=1$ הוא המינימלי וכן נגדיר $S[0]=0$)
כעת הבעיה שלנו הרבה יותר פתירה. ניצור מטריצה $n\times n$ (שתי מטריצות, אחת לכל שחקן) נסמנן B_1,B_2 ונמלא אותה בהתאם לפי נוסחת הנסיגה - $O(1)$ לכל תא. סה"כ n^2 תאים ומקום. סה"כ הפתרון המקסימלי עבור יעל יהיה $B_1[1,n]$.

$2018B$ בן גוריון שאלה 1

על לוח משחק ישנה ערימה סדורה של n מטבעות זהב. לכל מטבע יש ערך אי שלילי. נסמן ערך מטבע i בv_i.
במשחק משתתף שחקן יחיד, חוקי המשחק קובעים כי בכל תור על השחקן לאסוף $1,2,3$ ממטבעות הזהב בראש הערימה (לא יותר ולא פחות). ואז להשליך מטבע אחד נוסף מראש הערימה לפח. המשחק מסתיים כאשר לא נותרים איברים בערימה (האחרון נאסף או הושלך לפח). מטרת השחקן היא לאסוף מטבעות בערך מצטבר מקסימלי.
נתון כי ערכי המטבעות בערימה הם $\{v_1,...v_n\}$ בסדר הזה כאשר v_1 הוא המטבע הראשון בערימה והראשון שייאסף.
פתרון חוקי: מיקומי המטבעות אותם ניתן לאסוף
ערך פתרון: סכום ערכי מטבעות שנאספו.
א. נגדיר $sol(k)$ כאשר $0\le k\le n$ להיות קבוצת כל הפתרונות החוקיים עבור k המטבעות התחתונים בערימה.
הגדירו $OPT(k)$ והציעו נוסחה לחישוב.
ב. הצע אלגוריתם תכנון דינמי לפתרון הבעיה.
פתרון:
נגדיר $OPT(k)$ להיות ערך הפתרון האופטימלי עבור k מטבעות תחתונים בערימה. נראה כי אם $k=1$ אזי נוכל רק להחזיר v_1 ובדומה עבור $k=2,3$. החל מ$k>3$ נוכל להחליט האם לקחת את האיבר או להשליכו. נגדיר פורמלית כעת:

$$OPT(k)=\begin{cases} 0 & k=0 \\ v_n & k=1 \\ v_n+v_{n-1} & k=2 \\ v_n+v_{n-1}+v_{n-2} & k=3 \\ max\{OPT(k-2)+v_k, OPT(k-3)+v_k+v_{k+1}, OPT(k-4)+v_k+v_{k+1}+v_{k+2} & o.w \end{cases}$$

ב. מערך - זמן מילוי $O(n)$, אפשר להוריד מקום ל$O(1)$ עם שמירת מס' ערכים קודמים.

תרגיל 9 5202 - שאלה 1

הבעיה: נתון מערך A של מספרים (חלקים חיוביים וחלקם שליליים). מצאו זוג אינדקסים $i\le j$ שימקסם את הסכום $\sum_{k=i}^{j}A[k]$.
פתרון:
נתחיל מהאלגוריתם הנאיבי - נרצה לבחור בכל פעם 2 אינדקסים מתוך הסכום n כאשר הסדר לא משנה וגם חשיבות הבחירה לא משנה.
זה יעלה

$$\sum_{i=0}^{n-1}\sum_{j=i}^{n-1}1=\sum_{i=0}^{n-1}n-1-i=n-1-\sum_{i=0}^{n-1}i=n-1-\frac{n(n-1)}{2}=\frac{2(n-1)-n(n-1)}{2}=O(n^2)$$

עבור כל אינדקסים נצטרך לחשב את העלות $O(j-i) < O(n)$. ולכן סה"כ הפתרון הנאיבי הוא ב$O(n^3)$.
נעבור לפתרון הרקורסיבי: נגדיר $f(i,j)$ כפונקציה שמחשבת את הערך המקסימלי ביותר במקטע $[i,j]$. נגדיר את הפונקציה באמצעות נוסחת הנסיגה הבאה:

$$f(i,j) = \left\{ \begin{array}{cc} a_i & i=j \\ f[i,j-1]+a_j & o.w \end{array} \right\}$$

נראה כי נוכל לשמור את הנתונים בתוך מטריצה $n \times n$. כיצד נמלא אותה? נראה כי אם $i=j$ נרצה a_i ולכן נתחיל ממילוי האלכסון. לאחר מכן - נראה כי מתחת לאלכסון האברים לא רלוונטים כי $i \geq j$ ולכן המטריצה משולשית עליונה, מכאן שנראה שיש תלות בכל מילוי במילוי בעמודה הקודמת, לכן נמלא את המטריצה לפי עמודות. סה"כ מילוי המטריצה יעלה $O(n^2)$. כעת, נסרוק את המטריצה ונחפש את הערך הגדול ביותר, אותו ערך אנחנו נחזיר למשתמש, סריקת המטריצה תעלה $O(n^2)$ ולכן סה"כ פתרנו את הבעיה עם $O(n^2)$ מקום ו$O(n^2)$ זמן.
הוכחת נכונות לנוסחה: טענה - בהינתן זוג אינדקסים $i \geq j$, מתקיים כי אורך המקטע $[i,j]=[i,j-1]+a_j$.
הוכחה: נשמע טריוואלי אך נוכיח זאת בכל מקרה. שקול להראות $[i,j]-[i,j-1]=a_j$ ולכן

$$[i,j]=a_i+a_{i+1}+....+a_j$$

$$[i,j-1]=a_i+a_{i+1}+...+a_{j-1}$$

$$a_i+a_{i+1}+....+a_j-(a_i+a_{i+1}+...+a_{j-1})=a_j$$

כנדרש.
שחזור - קל, נחזיר את האינדקסים בהם היה האיבר המקסימלי ביותר.

תרגיל 9 5202 - שאלה 2

הבעיה: במפעל מכוניות כל מכונית צריכה לעבור בn תחנות לפני שהיא מוכנה. המפעל מחזיק בשני פסי ייצור מקבילים אשר כל אחד מורכב מn התחנות הללו. נסמן את התחנות בפס הראשון כ$a_{1,1},...a_{1,n}$ואת התחנות בפס השני כ $a_{2,1},........,a_{2,n}$. עבור כל פס ייצור נתון לנו הזמן שלוקח להעביר מכונית מכל תחנה לתחנה הבאה במסלול (המחירים לא זהים בשני המסלולים. כלומר להעביר מכונית מתחנה 2 ל3 במסלול 1, לא זהה להעברה של מכונית מתחנה 2 ל3 במסלול 2). מדי פעם מתקבלת הזמנה דחופה של מכוניות, ואז משתמשים בשני פסי הייצור בו זמנית על מנת ליצור מכונית אחת. הסיבה לכך היא שייתכן שיותר מהיר להעביר את המכונית מתחנה $a_{1,i}$ אל $a_{2,i+1}$ מאשר אל $a_{1,i+1}$. לצורך כך גם נתונים לנו מחירי העברה מכל תחנה לתחנה הבאה בפס האחר. מעבר מ$a_{1,i}$ ל$a_{1,i+1}$ יסומן כ$S_{1,i}$. כלומר העברה מ1 ל2 בפס הראשון היא במחיר $S_{1,1}$. בדומה עבור פס הייצור השני $S_{2,i}$ וכו'. המחיר לכניסה לפס הייצור לתחנה 1 יסומן e_1 והמחיר לכניסה לפס הייצור לתחנה 2 יסומן e_2. כמו כן, המחיר ליציאה מפס ייצור מסווים יסומן בהתאמה x_1,x_2. תארו אלגוריתם תכנון דינמי למציאת המסלול הייצור המהיר ביותר (לא להבהל, זו שאלה מאלגו' 1)
פתרון:
נתחיל מהפתרון הנאיבי - הנאיבי יהיה לעבור על כל האפשרויות הקיימות, תוך כדי לחשב את הזמנים ובסוף להחזיר את זמן הייצור הקטן ביותר שמצאנו. נראה כי בהינתן שאנחנו בתחנה יש שתי אפשרויות: ללכת לתחנה הבאה במסלול שלי או ללכת לתחנה הבאה במסלול האחר, לכן סה"כ קיימים 2^n אפשרויות. ולכן מעבר על כל הפתרונות ומציאת המינימלי יעלה $O(2^n)$. אקספוננציאלי ולא בא בחשבון.
נעבור לפתרון הרקורסיבי - נגדיר את הפונקציה הרקורסיבית $f(i,m)$ כאשר m הוא התחנה בה אני נמצא ויקיים $1 \leq m \leq n$ וכן i יציין את המסלול בו אני נמצא, ולכן $i \in \{1,2\}$. הפונקציה f תחזיר את זמן הייצור הנמוך ביותר, כאשר אנחנו נמצאים בתת בעיה כוללת של התחנות $1,..m$. נעיר כי נגדיר $m=0$ ככניסה למסלול. נראה כי יש שתי אפשרויות - להשאר במסלול או לעבור למסלול אחר
נגדיר את נוסחת הנסיגה להלן:

$$f(i,m)=\left\{\begin{array}{cc} e_1 & i=1\wedge m=1 \\ e_2 & i=2\wedge m=1 \\ min\{f(1,m-1)+s_{1,m-1},f(1,m-1)+t_{1,m-1}) & i=1\wedge 2\le m\le n \\ min\{f(2,m-1)+s_{2,m-1},f(2,m-1)+t_{2,m-1}) & i=2\wedge 2\le m\le n \end{array}\right\}$$

הוכחת נכונות: נוכיח כעת את נכונות הנוסחה. יש שני מקרי בסיס. הראשון הוא כאשר אנחנו רוצים לחשב את הזמן כאשר $i=1$ וכן $m=1$, במצב זה אני נמצא בתחנה 1 וכן במסלול 1, סה"כ הזמן עד כה הוא הזמן שלקח לי להכנס לתחנה, ולכן e_1. עבור e_2 ־ בדומה לחלוטין. אחרת, נדבר על המקרה בו $i=1$ וכן $n\ge m\ge 2$, במצב זה אנחנו בתחנה כללית כלשהי לאורך המסלול, וכן אנחנו בפס ייצור אחד. בפנינו שתי אפשרויות: הראשונה היא להשאר בפס הייצור הנוכחי, נרצה לשם כך את הזמן שלקח עד כה שזה $f(1,m-1)$ וכן את הזמן שיקח כעת לעבור מתחנה $m-1$ אל תחנה m. לכן סה"כ במקרה זה נחזיר $f(1,m-1)+S_{1,m-1}$. מקרה שני הוא כזה שבו נרצה לעבור מתחנה 1 לתחנה 2 ־ במקרה זה נרצה לחשב את הזמן של $f(1,m-1)$ שכן זה זמן הייצור המינימלי בתחנה 1 עד ל־$m-1$ שכן זה המצב בו אני נמצא כרגע, ועוד הזמן שאזדקק בשביל לעבור מתחנה $m-1$ אל תחנה m, בין מסלול 1 למסלול 2 שזה $t_{1,m-1}$. סה"כ המקרה השני כאשר $i=2$ סימטרי ואין צורך לדון בו.

פתרון תכנון דינמי: נשמור את הנתונים במטריצה שמימדיה יהיו $2\times n$. כאשר השורות ייצגו לי את הפסים והעמודות את התחנות. נתחיל ממילוי העמודה הראשונה בה נמלא את מקרי הבסיס עם e_1,e_2. נמלא את שאר העמודות, נראה כי בכל פעם המילוי יעלה $O(1)$ שכן מדובר בהכרעת מינימום ביחס לשתי תאים אחרים עם תוספות שידועות מראש, הפתרון שנרצה להחזיר הוא $min\{f(1,n)+x_1,f(2,n)+x_2\}$ שכן חישבנו את כל המסלול עבור כל אחת, ונרצה את הזמן המינימלי מבין שתיהם בסיום. זמן הסיבוכיות יהיה בזמן $2*n=O(n)$ שכן זה הזמן הנדרש למלא את המטריצה וגם במקום $O(n)$ שכן אחסנו מטריצה בגודל הזה $2\times n$. באשר לשחזור הפתרון ־ נתחיל מהמינימום שהוחזר לנו ונבדוק לאיזה מהתאים הוא מתאים, משם נבדוק האם הפתרון שקיבלנו הגיע כתוצאה מהמסלול הנוכחי ועוד המעבר או מהמסלול השני ועוד המעבר, וכן נתקדם בחזור, שחזור יעלה לנו $O(n)$. בהינתן כי לא נרצה שחזור נוכל לצמצם את המקום ולשמור בכל פעם את העמודה האחרונה לפניי, שכן לפניה לא רלוונטי מבחינתי. על פסודו נוותר.

תרגיל 5 3202 ־ שאלה 3

השאלה: נרצה מבנה נתונים יעיל שתומך בתחזוקה של מספרים. כאשר מבקשים מאיתנו לייצר את המבנה (בפעולת אתחול או בניה) נותנים לנו פרמטר m שמובטח שכל המספרים במבנה יהיו מספריים טבעיים בתחום $[1,m]$. הפרמטר לא משתנה לאחר הבנייה. הצע מבנה שתומך בפעולות הבאות:

1. אתחול מבנה ריק בזמן $O(1)$
ב. בניית מבנה (S,m) על קבוצה S בגודל $\sqrt{m}<n<m$ בזמן $O(n)$
3. הכנסת איבר חדש בזמן $O(logn)$
4. הוצאת איבר קיים בזמן $O(logn)$
5. חיפוש בזמן $O(logn)$.

פתרון: נשתמש בעץ AVL עם קצת סוויץ'. אתחול יהיה פעולה בסיסית, ניצור עץ AVL ריק ב$O(1)$. בנייה ־ בהינתן הקבוצה כיוון שנתון כי איברי S חסומים בתחום מסווים, נוכל למיינם במיון בסיס שיעלה $O(n)$. לאחר מכן, נוכל להשתמש בעובדה שהקבוצה ממוינת בכדי ליצור ממנה עץ AVL. נבחר את האיבר האמצעי ביותר במערך ־ הוא יהיה שורש העץ, משם נפעל ברקורסיה על שני החצאים וככה נבנה את העץ מהחצאיים רמה רמה. נטען כי $h(n)=\lceil log(n+1)\rceil-1$. טענה זו יש להראות באינדוקציה:

בסיס: $n=1$, $h(1)=\lceil log(2)\rceil-1=0$ ואכן בעץ עם קודקוד אחד גובהו 0.

צעד: נניח שנכון עבור n ונחלק למקרים.

א. n אי זוגי : אזי יש לנו חציון ואז בכל צד בעץ ישנם $\frac{n-1}{2}$

$h(n)=1+h(\frac{n-1}{2})=1+\lceil log((\frac{n-1}{2})+1)\rceil-1=\lceil log(\frac{n+1}{2})\rceil=\lceil log(n+1)\rceil-1$ כנדרש.

ב. אם n זוגי: אזי יש לנו שני עצים בגובה שונה. תת העץ הגדול (בה"כ T_1) מכיל $\frac{n}{2}$ערכים.

מכאן שגובה העץ $h(n)=1+h(\frac{n}{2})=1+\lceil log(\frac{n}{2}+1)\rceil-1=\lceil log((\frac{n+2}{2})\rceil=\lceil log(n+2)\rceil-1$

נשים לב שלכל מספר זוגי $x>2$ מתקיים:$\lceil logx\rceil=\lceil log(x-1)\rceil$

ולכן הוכחנו את הדרוש.

כעת נסתכל על קודקוד כלשהו בעץ המתקבל, עלינו להראות שהוא מקיים תכונת AVL. הקודקוד הזה עלה למקומו בהיותו חציון לתת מערך כלשהו.

*אם תת המערך היה אי זוגי $2k+1$ איברים אז בכל אחד מתתי העצים יש k ערכים ולכן הפרש הגבהים הוא בדיוק 0 ־ כדרוש.

* אם תת המערך היה זוגי $2k$ איברים אז בתתי העצים יש $k-1$ וk ערכים ולפי האינדוקציה שהראינו

$|h(k)-h(k-1)|=|\lceil log(k+1)\rceil-\lceil logk\rceil|\le 1$

סה"כ אכן קיבלנו שבנינו עץ AVL. סיבוכיות הזמן פועלת לטובתנו - בנינו זאת כמובן ב$O(n)$ שכן

$$T(n) = 2T(\frac{n}{2}) + O(1) = n^{log_2 2} = O(n)$$

לכן המיון n ועוד הבנייה n, $2n = O(n)$ לבנייה.
מכאן זה קל, חיפוש הוצאה והכנסה למבנה כפי שראינו בהרצאה זה ב$O(logn)$, הרי מדובר בעץ AVL רגיל לחלוטין, רק הבנייה שלו הייתה שונה.

שאלה חשובה!!!!!!

השאלה: נתונות שתי קבוצות של n נקודות, קבוצה אחת $\{p_1,..p_n\}$ נמצאות על הישר $y = 0$. והקבוצה השנייה $\{q_1,...q_n\}$ נמצאות על הישר $y = 1$. הקבוצות יוצרות n מקטעים ישרים ע"י חיבור כל נקודה p_i עם הנקודה התואמת לה q_i. כתבו אלגוריתם בשיטת הפרד ומשול שרץ בזמן $O(nlogn)$ ומחזיר את כמות החיתוכים בין זוגות של המקטעים הישרים שיצרנו (אם נקודה מסויימת היא חיתוך של יותר משני קטעים אזי היא נספרת כמס' זוגות הקטעים שנחתכים בה. למשל אם נקודה היא נק חיתוך של שני זוגות ישרים, אזי 2 נקודות חיתוך)

פתרון: הגישה הנאיבית היא לחשב את n משוואות הישרים, ואז לעבור עבור כל משוואה עם שאר המשוואות ולבדוק האם לה ולמשוואה אחרת יש נקודות חיתוך. לחשב נקודת חיתוך זה אמנם קוד שלוקח הרבה שורות אבל מתבצע ב$O(1)$ נניח. לכן סה"כ הנאיבי יעלה $O(n^2)$.

נראה כי עבור כל שני ישרים $p_i - q_i$ וכן $p_j - q_j$ נחתכים אמ"מ:

1. p_i נמצא משמאל לp_j אבל q_i נמצא מימין לq_j.
2. p_i נמצא מימין לp_j אבל q_i נמצא משמאל לp_j.

כלומר - ישנו היפוך בערכים. נכניס את ערכי הקבוצות למערכים. כעת הבעיה הופכת להיות מציאת כמות חילופי הסדר במערך: איך נפתור אותה? בדומה למרג' סורט. נפצל את המערך לשניים, נמיין ולבסוף נגיע למערך בגודל אחד ממויין. כעת במיזוג בחזרה לאחור, בכל שלב נבדוק האם איבר במערך הימני קטן מאיבר במערך השמאלי, ואז זה חילוף סדר. במצב זה כיוון שמיינו, מס' חילופי הסדר הוא בדיוק מס' האיברים במערך השמאלי , נסכום כך את מס' חילופי הסדר בעזרת מונה, סה"כ סיבוכיות זהה למרג' סורט וזה $nlogn$.

1202 תרגיל 3 שאלה 3

הציעו מבנה נתונים דמוי מחסנית , מחסנית מינימום, התומך בפעולות הבאות: אתחול, האם ריק,הכנסה לראש,הוצאת האיבר בראש והחזרתו,וגם: החזרת איבר המינימום (ללא הוצאה). הצע מבנה נתונים שמשתמש במיקום לינארי במס' האיברים במחסנית ומבצע כל פעולה בזמן קבוע (נתחו לשיעורין)

פתרון: נשתמש במחסנית רגילה וחוץ ממנה במחסנית מינימום. באתחול נאתחל בנוסף גם מחסנית נוספת בשם s_2 שתהיה לנו לעזר. כאשר נכניס איברים אל המבנה, נכניס אותם ובנוסף נשאל האם הם קטנים מאיבר המינימום שנמצא כעת בראש המחסנית s_2 (בהתחלה בפעם הראשונה יוכנס גם לשם כל איבר). אם כן, נדחוף אותו אל המחסנית s_2 שכן הוא כעת האיבר המינימלי במחסנית. האם ריק זה קלאסי אותו דבר. ובאשר לאתחול כבר דיברנו. גם על הכנסה. מה שנותר זה לדבר על החזרת איבר המינימום והוצאה של האיבר בראש. ובכן החזרת איבר המינימום זה פשוט $returns_2.top$ שזה $O(1)$. כעת באשר להוצאה מהמחסנית - כאשר נוציא איבר מהמחסנית נוציא גם את האיבר שבראש המחסנית s_2 , וזה יבטיח שנקבל את המינימום שהיה לפני שנכנס אותו איבר למחסנית. (כלומר בדחיפה, אם למשל נרצה לדחוף 5 למחסנית שהמינימום בה הוא 2, המינמום ישאר 2 אבל נדחוף 2 שוב, מקווה שברור ההסבר). כעת נותר לנתח - ננתח לפי שיטת הבנק. ב

האם ריק: נשלם מטבע אחד בדיוק, פעולה בסיסית שהיא $O(1)$. אתחול בדומה.
הכנסה לראש: נשלם שתי מטבעות, אחת להכנסה למחסנית הרגילה, השנייה להכנסה למחסנית המינימום
הוצאה של איבר: נשלם 2 מטבעות, אחת להוצאה ממחסנית רגילה והשנייה להוצאה ממחסנית מינימום
החזרת מינימום: נשלם מטבע אחד, שהרי מדובר בפעולה בסיסית של $return$.
סה"כ תמיד הכל קבוע ב$O(1)$.

2021 תרגיל 5 שאלה 3

תארו מבנה נתונים המאחסן קבוצה של n מספרים ותומך ב:

1. בנייה - מקבלת מערך של n איברים ובונה מבנה ב$O(n)$
2. שאילתה: מקבלת מס' $1 \leq k \leq n$ ומחזירה את האיבר הk בגודלו בקבוצת המספרים בזמן $O(n)$.

פתרון: נשתמש במערך שקיבלנו. נשתמש באלגוריתם סלקט ונפעיל אותו על חציון החציונים. כעת נקבל מערך בו חצי מימין גדולים ממני וחצי משמאל קטנים. נבדוק האם האיבר האמצעי קטן או גדול מk ונלך רקורסיבית לצד המתאים. כך נמשיך עד שנמצא את האיבר הk בגודלו. נראה כי

$$T(n) = 2T(\frac{n}{2}) + O(1) = O(n)$$

כמו כן מציאת חציון החציונים עם אלגוריתם סלקט זה $O(n)$.

2025 איילת מועד א'

שאלה 1:

נרצה לממש מבנה נתונים מילוני בעזרת **עץ** התומך בפעולות:

א. הכנסה - מכניס לעץ ביעילות של $O(1)$ לשיעורין

ב. חיפוש - מחפש בעץ ביעילות של $O(logn)$ לשיעורין.

ידוע כי הכנסת איברים נעשית רק בסדר עולה.

פתרון:

נרצה להשתמש בעץ AVL.

כך נקבל את החיפוש בעץ ממש במקרה הגרוע ביותר ב$O(logn)$ כאשר n הוא מס' האיברים בעץ, ובפרט לשיעורין נקבל $O(logn)$.

כעת נותר להסביר מדוע הכנסה אל העץ תעלה $O(1)$ לשיעורין. נראה כי קיבלנו נתון מעניין והוא שהכנסת איברים אל העץ מתבצעת רק בסדר עולה. הדבר היה שקול אם היינו מקבלים מערך ממויין ומנסים לבנות ממנו עץ - הוכחנו בתרגול שניתן לעשות זאת ברקורסיה על החציון ושני החצאים ב$O(n)$ עבור n איברים. לכן באופן שקול מה שנעשה יהיה לתת מטבע אחד בדיוק לכל איבר בכניסה. נשים לב כי כאשר איבר יכנס אל העץ, בפרט יכנס איבר גדול ממנו בכל שלב ולכן נצטרך א. להכניס אותו אל העץ, 2. לבצע מס' קבוע של פעולות החלפה שחסומות בc קבוע על מנת לסדר את העץ שיהיה עץ AVL. כיוון שהאיברים מתבצעים בסדר עולה אנחנו לא צריכים לבצע תזוזות בגובה העץ, ולכן אם לכל איבר נקצה בכניסתו אל המבנה $1+c$ מטבעות, נראה כי לשיעורין $O(1+c) = O(1)$ שיעלה לבצע הכנסה אל תוך העץ.

מדוע מס' ההחלפות יהיה קבוע?

נשים לב כי כשמכניסים איברים בסדר עולה כל איבר יכנס כבן ימני של האיבר הקודם, אם נבצע גלגול השפעת הכנסה על הגלגול כעת תהיה מס' קבוע של פעולות. אנחנו נסמן את מס' הפעולות הקבוע הזה c (המרצה טליה טענה שאיננו צריכים לזכור בע"פ גלגולים), ולכן לשיעורין יתבצעו $c+1$ החלפות קבועות, ולכן סה"כ לשיעורין $O(1)$ להכנסה אל תוך העץ.

שאלה 2:

בהינתן מחרוזת S של אותיות קטנות באנגלית, נגדיר את ערך המחרוזת כסכום הריבועים של תדירות כל תו ייחודי במחרוזת. כלומר, אם נתונה המחרוזת $S = character$ התדירות של האותיות במחרוזת S היא: $a = 2, c = 2, e = 1, h = 1, r = 2, t = 1$ ולכן ערך המחרוזת הוא $15 = 2^2 + 2^2 + 1^2 + 1^2 + 2^2 + 1^2$.

להלן בעיית צמצום ערך מחרוזת תלוי k:

קלט: מחרוזת S של אותיות קטנות באנגלית ומס' k קטן מאורך המחרוזת

פלט: מחרוזת S' עם ערך מחרוזת מינימלי שמקבלת מהמחרוזת S לאחר הורדה של k אותיות.

כלומר עבור הדוגמה מקודם אם נקח $k = 2$ נוכל להוריד את האותיות a, c וכך ערך המחרוזת ירד ל9.

כתוב אלגוריתם לפתרון הבעיה בזמן $O(n + min\{klogn, klogk\})$. רמז - העזר במבנה נתונים לניהול הקלט.

פתרון:

ננסה להבין מה קורה כאן מבחינת סיבוכיות. נדרש מאיתנו לבצע מעבר לינארי על הקלט ולאחריו עוד מינימום מבין שני אפשרויות - יתכן ריצה במקביל? עו נשקול זאת. נראה כי יש כאן $klogn$ ו$klogk$. מזכיר ערימה יחסית. אם עושים ערימה של כל האיברים גובהה יהיה $logn$. אם עושים ערימה של k פוטנציאלים להורדה גובהה יהיה $logk$.

תובנה שנרצה לשים לב אליה - בשביל לקבל ערך מחרוזת מינימלי, נרצה להוריד קודם כל את האיברים עם התדירות הכי גבוהה. כיוון שערך המחרוזת מורכב מהעלאה בריבוע של מספרים. נרצה לדעת לכל איבר את התדירות שלו ולכן נרצה לדעת כמה איברים שונים יש קודם כל. לשם כך, נכניס את האיברים לתוך טבלת האש, בכל פעם בכניסה אליה אנחנו נבדוק אם האיבר לא קיים נגדיל קאונטר ואם קיים נכניסו מבלי להגדיל קאונטר. כעת, אנחנו יודעים מס' איברים ששונים זה מזה. נסמן ערך זה כx. בהכרח $x \leq n$. נבנה מערך עזר בגודל n. כעת נוציא חזרה את האיברים מטבלת האש חזרה למחרוזת או שפשוט נשתמש במחרוזת המקורית. כעת נבנה טבלת תדירויות: נדאג שכל איבר יהיה עם פוינטר למיקומו בטבלת השכיחויות. נעבור על המחרוזת משמאל לימין. נקח תו, אם התו הזה כבר קיים בתוך טבלת האש שלנו, אזי נלך עם הפוינטר אל מיקומו בטבלת השכיחויות ונגדיל את ערך השכיחות ב1. אחרת, נפתח עבורו תא חדש בתוך המערך, נכניס את האיבר לטבלת האש עם פוינטר לקאונטר במערך וכמובן נגדיל קאונטר לאחד.

סה"כ עד כה קיבלנו טבלת שכיחויות ב$O(n)$.

כעת, נבנה ממערך השכיחויות ערימה. כמובן שכל השכיחויות עדיין מקושרות לתוך הערך אליהם הן שייכות. סה"כ בידינו כרגע ערימה בגודל x שנבנתה ממערך בגודל x ולכן עלות בניית הערימה $O(x)$. נבהיר כי הערך בערימה הוא לפי השכיחות של התו וכן הוא מקושר באמצעות פוינטר.

כעת מה שנרצה לעשות יהיה להתייחס לתכונת המינימום - נרצה לבצע $min\{klogn, klogk\}$ הוצאות שונות. ננסה לחשוב על זה. המטרה היא להוציא k ערכים מהמחרוזת. כמובן נעיר כי אנחנו עם ערימת מקסימום שכן המטרה להוציא ערך גדול ביותר. מה שנרצה בכל שלב יהיה לקחת את איבר המקסימום - להוציא אותו מהערימה, לעדכן בטבלת השכיחות שגודלו ירד ב1, וכעת להחזירו אל הערימה (כמובן שערכו ירד ולכן נדרש להוציא ולהכניס). סה"כ בוצעה הוצאה והכנסה מחדש ולכן זה עלה $2logx = O(logn)$. סה"כ ביצענו k הוצאות כאלו (רצינו להוריד ערך בk מילים) ולכן זה עלה $klogn$. מיד נחזור לדרישת המינימום.

כעת נסביר כיצד בהינתן טבלת השכיחויות החדשה המעודכנת ניתן לכתוב את המחרוזת מחדש. שמרנו את ערך המחרוזת הקודמת, וכן שמרנו לכל איבר את השכיחויות. מה שנעשה יהיה מעבר לינארי על המחרוזת שכן כעת נחליט שכאשר אנחנו הורדנו שכיחות מסוימת, סימנו זאת עם איזשהו שדה של כמה איברים הוא צריך למחוק מעצמו (קל לחשב זה ההפרש). כעת נעבור על איברי המחרוזת, נניח והגענו לאיבר s. ובשדה שלו כתוב מס' שונה מאפס, אזי נמחוק אותו מהמחרוזת. וכך נתקדם לינארית על המחרוזת. כמובן אם נתקל שוב בs למשל והשדה שלה איננו אפס נמחק גם אותה. סה"כ לבסוף קיבלנו מחרוזת עם ערך מינימלי. מעבר לינארי זה עלה $O(n)$

כעת נשוב להסביר על המינימום $min\{klogn, klogk\}$, הראינו כיצד לבצע ב$klogn$, כעת אנחנו נדרשים להסביר כיצד נבצע ב$klogk$. ניצור ערימת עזר בגובה k. אנחנו יודעים שk האיברים עם השכיחות המקסימלית יהיו לכל היותר עד גובה k בערימה המקורית. ולכן ניצור ערימת עזר של כל האיברים עד לרמה k. כעת הפתרון יהיה זהה לכמו קודם בדיוק רק שכעת גובה הערימה המדוברת הוא k. (הערה - ראינו בהרצאה כי k האיברים הגדולים ביותר נמצאים לכל היותר עד רמה k בעץ)

סה"כ זו ריצה במקביל על שתי האפשרויות ונפסיק כאשר נקבל פתרון כלשהו ולכן זה יהיה ב$min\{klogn, klogk\}$ וכן מעברים לינאריים על הקלט ולכן סה"כ $O(n + min\{klogn, klogk\})$. כנדרש.

שאלה 3:

כתוב אלגוריתם לינארי (זמן ומקום $O(n)$ אם נדרשים למקום) לפתרון הבעיה הבאה:

קלט: מערך של מס' שלמים ומס' $k = O(n)$

פלט: אמת אם ניתן לחלק את המערך לזוגות, כך שסכום כל זוג מתחלק בk ללא שארית.

פתרון:

נרצה למצוא בכל שלב a, b שיקיימו $a + b = ck$ כאשר c טבעי. לפתור ב$O(n^2)$ היינו עושים עם זוגות והאש. אבל זה לא דרישת הסיבוכיות שלנו.

תובנה: או ששניהם מתחלקים בk, ואז בפרט סכומם מתחלק בk. או שסכומם יתחלק בk. כמו כן, אם אחד מהם מתחלק בk והשני לא - בוודאות סכומם לא יתחלק בk. הוכחה:

אחד מתחלק בk כלומר קיים c טבעי כך ש$a = ck$. השני לא מתחלק בk. סה"כ נקבל $\frac{a+b}{k} = \frac{ck+b}{k} = c + \frac{b}{k}$ וb כאמור לא מתחלק בk.

לכן כעת קיבלנו תובנה מעניינת:

א. אם שניהם מתחלקים בk, גם סכומם יתחלק בk.

ב. אם אחד מתחלק בk והשני לא, סכומם בוודאות לא יתחלק בk.

ג. אחרת, אם אף אחד מהם לא מתחלק בk, **ייתכן** כי סכומם בכל זאת מתחלק בk. מתי זה יקרה? נראה כי נניח ויש לנו שני מספרים שאינם מתחלקים בk. אך סכומם כן, כלומר $c = \frac{a+b}{k}$. כלומר, $a + b = ck$. מתי נקבל כי האיברים במצב כזה יתחלקו בk? אמ"מ סכום השאריות שלהם יהיה שווה בדיוק לk.

לכן מה שנעשה יהיה כזה: ניצור מערך B ונחשב בו לכל איבר את שארית החלוקה שלו בk. אנחנו יודעים שמערך כזה יקיים שאיבריו תחומים ב$[0, ...k - 1]$ וכן $k \in O(n)$ ולכן ניתן למיין אותו באמצעות מיון שאינו מבוסס השוואות ב$O(k) = O(n)$. כעת, יש לנו מערך שאריות של האיברים. נעבוד לפי המסקנה מלמעלה - תחילה נחבר את הזוגות שמתחלקים בk, כלומר אלו עם מקדם 0, אם קיבלנו שיש מס' אי זוגי של זוגות כאלו לצערנו זה אומר שאין חלוקה מתאימה כי כפי שראינו אם אחד מתחלק בk והשני לא יתחלק הסכום גם לא יתחלק. נניח וכן יש מס' זוגי של כאלו אז נתקדם בחלוקתם לזוגות. כעת, אנחנו מסתכלים על האיברים ששארית שלהם הינה $1, ...k - 1$. כעת אף אחד מהם לא מתחלק בk. לפי תובנה ג', סכומם יתחלק בk אמ"מ סכום השאריות יהיה k. שוב, אם בידינו מס' אי זוגי של איברים שנותרו (לאחר הורדת האיברים שמתאימים לשארית אפס), בוודאות ממילא אי אפשר לחלקם לזוגות לכן נחזיר שקר. אחרת, כעת יש מס' זוגי של איברים שנותרו. נבנה טבלת האש ונכניס אליה את האיברים הללו, בוודאות מס' האיברים כאן קטן מn. לכן גודלה יהיה $O(n)$. כעת נעבור על כל אחד מהאיברים במערך הקלט שנשאר, נניח ושארית החלוקה שלו היא r אזי נחפש בטבלת האש האם קיים $k - r$. אם כן, נוציא אותו ונסמן על האיבר הזה בשדה מסויים שכבר שיבצנו אותו ונתקדם הלאה (הבהרה - האיברים יהיו מקושרים בפוינטרים)

ביותר פירוט - לכל איבר מהאיברים שנותרו נוסיף שדה האם לקחתי אותו או שלא וכן נבנה טבלת האש שתכיל פוינטרים לאיברים הספציפיים עם הערכים. נעבור לינארית על המערך שנותר ונחפש האם בהינתן שאני בשארית r האם קיים $k - r$. אם כן נגש אליו נסמן אצלו שלקחנו אותו ונתקדם הלאה ברשימה. סה"כ זה יהיה מעבר לינארי על האיברים, בשילוב בניית המערך מקודם נקבל $O(n)$ לכל הבעיה עם $O(n)$ זמן.

שאלה 4:

הגדרה: בהינתן מערך A של מספרים שלמים בגודל n נאמר כי במערך A קיים היפוך אם קיימים שני איברים במערך כך ש-$i < j$ אך $A[i] > A[j]$.
קלט: מערך A של מס' שלמים בגודל n.
פלט: מס' ההיפוכים במערך.
הערה: אם המערך ממויין כבר בסדר עולה מס' ההיפוכים יהיה 0. אם מערך ממויין בסדר יורד, מס' היפוכים יהיה מקסימלי.
הצע אלגוריתם יעיל ככל הניתן לפתרון הבעיה
פתרון:
כעת נטען שהאלגוריתם חייב לרוץ בזמן $O(nlogn)$, אחרת יכולנו למיין באמצעות שיטה זו וזה היה בסתירה לחסם תחתון.
ניתן לפתור ב-$O(n^2)$ נאיבי עם שתי לולאות, עבור כל איבר לבדוק את כל איבריו ולבדוק האם מתקיים הטענה מלמעלה ולהוסיף לקאונטר - סה"כ $O(n^2)$.
נרצה כמובן לשפר - מתי מקבלים $nlogn$? או שממיינים, מה שלא נרצה לעשות כי זה יהרוס לנו את האינדקסים הקיימים, או שבונים עץ AVL או ערימה ומהם מבצעים הוצאות והכנסות, ונראה שגם זה לא הכיוון, או שעובדים בהפרד ומשול ומגיעים לנוסחת הנסיגה $T(n) = 2T(\frac{n}{2}) + O(n)$ שפתרונה $O(nlogn)$.
נרצה להסתכל על האיבר האמצעי ביותר, שאינדקסו $\frac{n}{2}$. נרצה לחלק את הבעיה לשתי בעיות: מצא לי כמות היפוכים בימין, מצא לי כמות היפוכים משמאל. מצא לי כמות חילופים שמתחילים משמאל ומתסיימים מימין. וחבר. כמה עולה למצוא כמות היפוכים של התחלה משמאל וסיום מימין? כמו שראינו בתרגול עם תת מערך כבד ביותר - $O(n)$
לכן -
הפרד: נפריד את המערך לשני חלקים ע"י בחירת האיבר האמצעי.
משול: פתור תת בעיה רקורסיבית, יעלה לך $O(n)$ וכן עד שתגיע לקלט של 2.
צרף: חבר את סכום החילופים מימין ומשמאל יחד וזה מס' החילופים במערך כולו.
כיצד נדע היפוך? כאשר איבר מהחלק הימני קטן מאיבר מהחלק השמאלי - כמו במרג' סורט, זה אומר שיש היפוך. נראה פסודו

```
function countInversions(arr, left, right):
  if left >= right:
    return 0
mid = (left + right) / 2
  leftInv = countInversions(arr, left, mid)
  rightInv = countInversions(arr, mid+1, right)
  crossInv = mergeAndCount(arr, left, mid, right)
  return leftInv + rightInv + crossInv

function mergeAndCount(arr, left, mid, right):
  leftArr = arr[left...mid]
  rightArr = arr[mid+1...right]
  invCount = 0
  i = 0, j = 0, k = left
  while i < leftArr.length AND j < rightArr.length:
    if leftArr[i] <= rightArr[j]:
      arr[k] = leftArr[i]
      i++
    else:
      arr[k] = rightArr[j]
      invCount += (leftArr.length - i)
      j++
    k++

  // העתק שאר איברים...

  return invCount
```

השורה המדוברת בפסודו היא $invCount+ = leftsize - i$. כאשר איבר מהמקטע הימני קטן מאיבר מהמקטע השמאלי, הוא קטן מכל האיברים שנשארו במקטע השמאלי (כי זה מיון הרי)

שאלה 5 אלגו' 1 שנת 2025

הגדרה: יהי רצף רנ"א $S = s_1, ..., s_n$. לכל i $s_i \in \{A, U, C, G\}$. זיווג רנא של S הוא קבוצת זוגות של אינדקסים שיקיימו את התנאים הבאים

א. כל אינדקס נמצא בזוג אחד לכל היותר
ב. כל זוג (i, j) יקיים $i < j$
ג. כל זוג צריך לקיים $(s_i, s_j) \in \{AU, UA, CG, GC\}$.
4. אם קיים זוג (i, j) אזי לא קיים זוג (x, y) כך ש$i < x < j$ ו$y \notin [i + 1, j - 1]$.

קלט: רצף S
פלט: גודל זיווג רנ"א גדול ביותר כלומר מס' זוגות גדול ביותר.
הבעיה ניתנת לפתרון בתכנון דינמי.

פתרון:

נרצה להגדיר נוסחת נסיגה. נגדיר $f(i, j)$ כמס' הזוגות הגדול ביותר שניתן ליצור מהאיברים $(i, ...j)$ בסדרה S וכמובן הפתרון לבעיה הגדולה יהיה ב$f(i, n)$. נראה כי בהינתן שבחרנו זוג, נרצה לעבור על כל $i \leq x < y \leq j$ בניהם. לכן נגדיר

$$f(i,j) := \left\{ \begin{array}{cc} 0 & i \geq j \\ 1 + f(i+1, j-1) & S[i], S[j] \in AU, UA, CG, GC\}\} \\ max_{1 \leq k \leq j-1}\{f(i,k) + f(k+1, j)\} & o.w \end{array} \right\}$$

הסבר: נרצה לעבוד רק כאשר $i < j$. בכל שלב בידינו שלוש אפשרויות: לוותר על i, לוותר על j או להסתכל על הזוגות בניהם ולמצוא מתוכם מקסימום. נשים לב שגם תנאי 4 מתקיימת אם בחרנו זוג אזי בהכרח הקריאה $f(x + 1, y - 1)$ תמנע הצטלבויות כדקלמן.

כעת ניצור מטריצה $n \times n$. משולשית עליונה. נמלא מתחת לאלכסון אפסים. נמלא אותה לפי נוסחת הנסיגה , נראה כי מילוי תא עשוי לקחת $O(n)$ זמן ולכן סיבוכיות הזמן הכוללת תהיה $O(n^3)$ זמן וכן $O(n^2)$ מקום - לא ניתן לצמצם את המקום כי אנחנו נדרשים לתאים קודמים בנוסחה לחישוב.

שאלה 2 2020 מועד מאי

הנוסע המתמיד פיליאס פוג נושא במזוודתו ממירים בין שקעי חשמל של מדינות שונות מהסוגים הבאים (כל ממיר מתואר בפורמט של מדינת כניסה $\longleftarrow$ מדינת יציאה)

א. ישראל $\longleftarrow$ בריטניה

ב. בריטניה $\longleftarrow$ סין

ג. סין $\longleftarrow$ בריטניה

ד. בריטניה $\longleftarrow$ ישראל

ה. סין $\longleftarrow$ סין.

ניתן לחבר ממיר מסוג א' לממיר מסוג ב' אם ורק אם סוג החיבור ביציאה מממיר א' זהה לסוג החיבור בכניסה של ממיר ב'

בבעזרת חיבורים כאלו ניתן ליצור שרשרת של ממירים.

שתי שרשראות ממירים מאותו אורך ייחשבו שונות אם קיים מיקום i כך שבמיקום הi יש ממיר מסוג שונה בכל שרשרת

הניחו שלתייר יש מס' גדול כרצונו של ממירים מכל אחד מהסוגים ברשימה.

תאר אלגוריתם תכנון דינמי יעיל אשר בהינתן מס' $n \in \mathbb{N}$ מחשב את מס' השרשראות השונות של ממירים שהתייר יכול ליצור מהסוגים הנ"ל שאורכן הוא בדיוק n.

פתרון:

נרצה ראשית ליצור דרך שבה אנחנו נשמור את המידע אודות 5 סוגי הממירים. לכן אנחנו נגדיר מטריצה בגודל 2×5 ונקרא לה B. בכל עמודה יהיה סוג ממיר. בשורה הראשונה יהיה נתיב הכניסה ובשורה השניה נתיב היציאה. נסמן את המדינות עם האותיות I, B, C (ישראל, בריטניה וסין). כעת נוכל לגשת בקלות יותר לבעיה באמצעות המטריצה - נעיר שחישוב מוקדם זה לא עלה לנו כלום כלומר $O(1)$ זמן כי המימדים קבועים וכן הערכים בהם. כעת, בהינתן מס' טבעי n נרצה לבדוק את מס' שרשראות שונות של ממירים. נגדיר $f(i,j)$ כמס' שרשראות באורך $1 \leq i \leq n$ שאפשר לעשות מi ממירים, כך שהממיר האחרון שבחרתי היה j שלו היו חמש אפשרויות. כעת נשים לב במקרים הבאים:

א. אם $i = 1$ כלומר זה הראשון שנכניס, אזי אנחנו לא יודעים היכן הוא היה קודם כי הוא רק התחיל - ולכן בפניו 5 אפשרויות כיצד להתקדם במסלול.

אחרת, $i > 1$. נרצה לפצל למקרים.

אם $j = 1$ אזי השתמשתי בבריטניה ביציאה, כעת אוכל לבחור רק מבין 2 ו4.

אם $j = 2$ אזי השתמשתי בסין ביציאה ולכן אוכל לבחור רק מבין 3 ו5

אם $j = 3$ השתמשתי ביציאה מבריטניה ואוכל רק מבין 2 ו4

אם $j = 4$ אזי השתמשתי ביציאה בישראל וכעת אוכל רק 1

אם $j = 5$ השתמשתי ביציאה מסין, אוכל לקחת את סין (5) או את 3.

כעת נביא זאת לידי ביטוי בנוסחת הנסיגה הבאה:

$$f(i,j) := \left\{ \begin{array}{cc} 1 & i = 1 \\ f(i-1,2) + f(i-1,4) & j = 1 \\ f(i-1,3) + f(i-1,5) & j = 2 \\ f(i-1,2) + f(i-1,4) & j = 3 \\ f(i-1,1) & j = 4 \\ f(i-1,5) + f(i-1,3) & j = 5 \end{array} \right\}$$

כעת נפתור בתכנון דינמי: ניצור מטריצה $n \times 5$ ונמלאה לפי עמודות ותנאי הבסיס. סה"כ זמן למילוי תא יעלה $O(1)$. נראה כי בכל תא במטריצה $A[i,j] = f(i,j)$. נראה כי מסלול יכל להסתיים בכל אחד מהתאים ולכן סה"כ הפתרון יהיה $\sum_{k=1}^{5} A[n,k]$. נוכל לשמור בכל שלב את העמודה הקודמת והנוכחית ולכן נקבל סיבוכיות זמן $O(n)$ ומקום $O(1)$!!!.

2018 **מועד ב' טופס שאלה 5 אלגו**

נתון מערך A של מספרים ממשיים $A = [a_1, ..., a_n]$. d חלוקה של המערך A היא בחירה של $d-1$ אינדקסים שמגדירים קטעים $[a_1, .a_{i_1}], ..., [a_{i_{d-1}+1}, .., a_n]$.

עלות מקטע בd חלוקה הוא סכום המספרים באותו מקטע.

עלות d חלוקה היא עלות מינימלית של מקטע מבין מקטעים שהיא מגדירה.

כתוב אלגוריתם תכנון דינמי שמקבל מערך A ומס' d חיובי ויחזיר את העלות המינימלית של d חלוקה.

פתרון:

ראשית נעשה עיבוד מקדים, נראה כי אנחנו נעבוד כאן עם מקטעים וסכומים ולכן נגדיר מערך סכומים S כאשר $S[i] = \sum_{k=1}^{i} A[k]$. נחשב אותו באמצעות ההגדרה הרקורסיבית הבאה $S[1] = a_1$ וכן לכל $i > 1$ יתקיים $S[i] = S[i-1] + a_i$

. סה"כ מילוי המערך כך יעלה $O(n)$ זמן ומקום. כעת אם נרצה למצוא את תת הסכום $\sum_{k=i}^{j} A[k] = S[j] - S[i-1]$ סה"כ חישוב ב$O(1)$ לאחר העיבוד המקדים.

כעת נגדיר פונקציה f אשר $f(i,k)$ תהיה העלות המינימלית של d חלוקה עבור המספרים $a_1, ..., a_i$ לk מקטעים כעת, נראה כי: אורך מקטע חייב להיות לפחות 1. ולכן בשביל לחלק לd קטעים נהיה חייבים לעבוד עם $i \geq d$. כאשר $i < d$ נגדיר בפונקציה אפס.

אחרת, $i \geq d$ ונרצה לבחור $d-1$ אינדקסים שונים לשים בהם את האיבר. נרצה לעבור על כל נתאר את נוסחת הנסיגה הבאה:

$$f(i,k) := \left\{ \begin{array}{cc} \infty & i < d \\ min_{1 \leq j \leq i-1}\{f(j,k-1), S[i] - S[j-1]\} & i \geq d \\ S[i] & k = 1 \end{array} \right\}$$

למעשה ־ נבחר נקודה לחלוקה, ניצור ממנה תת בעיה על $k-1$ מקטעים ונחשב את המינימום מבין לקחת את המקטע הספציפי הזה או לחשב את סכום תת המקטע הנ"ל.

נמיר את הבעיה לתכנון דינמי וניצור מטריצה בגודל $n \times d$. נמלאה לפי נוסחת הנסיגה ונראה כי מילוי כל תא עשוי לעלות $O(n)$ זמן ולכן סה"כ סיבוכיות הזמן $O(n^2 d)$ והמקום $O(nd)$.

תרגיל בית 4 שאלה 4

שאלה 4 ־ נשנה את הגדרת עץ AVL ונדרוש שמקדם האיזון יהיה מקסימום 2. נסמן ב־$T(h)$ את מס' הקודקודים המינימלי בעץ כזה בעומק h והוכיחו באינדוקציה על h כי $T(h) \geq h^{\frac{2}{3}} - 1$.

פתרון: יהי עץ AVL שמקדם האיזון בו הוא מקסימום 2. נוכיח באינדוקציה על גובה העץ h.

בסיס ־ $h = 0$ כלומר קודקוד בודד. מס' הקודקודים בעץ הוא 1 ־ כי קודקוד בודד, ואכן $T(h) = 1 > 0^{\frac{2}{3}} - 1 = -1$

צעד: נניח שהטענה נכונה עבור עץ AVL מגובה h ומטה באינדוקציה שלמה. כלומר $T(h) \geq h^{\frac{2}{3}} - 1$. נרצה להראות כי $T(h+1) \geq (h+1)^{\frac{2}{3}} - 1$.

יהי עץ AVL מגובה $h+1$. נתבונן על העץ הנ"ל ששורשו r. בה"כ נניח כי יש לו תת עץ שמאלי מגובה h, ותת עץ ימני מגובה מינימלי של $h-2$.

כעת, שני תתי העצים מקיימים את הנחת האינדוקציה. כלומר ־

$$T(h) \geq h^{\frac{2}{3}} - 1$$

$$T(h-2) \geq (h-2)^{\frac{2}{3}} - 1$$

מכאן, שמס' הקודקודים בעץ שלנו יהיה:

$$T(h+1) \geq 1 + T(h) + T(h-2) = 1 + h^{\frac{2}{3}} - 1 + (h-2)^{\frac{2}{3}} - 1 = h^{\frac{2}{3}} + (h-2)^{\frac{2}{3}} - 1$$

נרצה להוכיח כי

$$h^{\frac{2}{3}} + (h-2)^{\frac{2}{3}} - 1 \geq (h+1)^{\frac{2}{3}} - 1$$

כלומר

$$h^{\frac{2}{3}} + (h-2)^{\frac{2}{3}} \geq (h+1)^{\frac{2}{3}}$$

שקול

$$h^{\frac{2}{3}} + (h-2)^{\frac{2}{3}} \geq (h+1)^{\frac{2}{3}}$$

טענה זו נכונה לכל $h \geq 4$ (יש לגזור ולהראות שהפונקציה תמיד חיובית/שלילית....)

3202 תרגיל בית 4 שאלה 3

עץ m־ארי מושלם הינו עץ בו לכל קודקוד יש m בנים, או שהינו עלה. הוכיחו את הטענות הבאות בעזרת אינדוקציה על מבנה העץ (פתרון שלא יתבסס על כך לא יזכה במלוא הנקודות):

א. עבור $m > 1$ ואי־זוגי מתקיים כי לעץ m־ארי מושלם יש כמות אי ־זוגית של עלים.

ב. לעץ m־ארי מושלם עם m עלים יש$\frac{n-1}{m-1}$ קודקודים פנימיים.

פתרון:

א. נוכיח באינדוקציה על מבנה העץ ־ דהיינו גובה העץ.

בסיס: נוכיח עבור $h = 0$. במצב כזה יש לנו קודקוד בודד, כלומר הקודקוד הינו עלה. ואכן יש לנו סה"כ עלה 1 ־ כמות אי זוגית של עלים.

צעד: נניח נכונות עבור עץ m ארי מגובה h. נתבונן על עץ מגובה $h+1$. העץ מורכב מקודקוד הראש, נסמנו r ולו יש m בנים סה"כ. כל אחד מ־m העלים הוא תת עץ בעצמו מגובה קטן מ$h+1$ ולכן לפי הנחת האינדוקציה בכל אחד מהם יש מס' אי זוגי של עלים. מכאן שסה"כ נקבל שבעץ הגדול יש לנו m פעמים מס' אי זוגי של עלים ־ m הוא מס' אי זוגי וכן מכפלה של שני מספרים אי זוגיים תתן לנו מספר אי זוגי ולכן יש בעץ כמות עלים אי זוגית.

ב. נוכיח באינדוקציה שוב, על גובה העץ.

בסיס: יהי איזשהו $m > 1$ כך שאכן $m-1 > 0$. נתבונן על המצב $n = 1$, קודקוד בודד. אכן אין לו קודקודים פנימיים.

צעד: נניח שהטענה נכונה לכל עץ עד $n-1$ עלים. נוכיח זאת עבור עץ עם n עלים. יהי עץ m ארי מושלם עם n עלים. נסמן את ילדיו ב־$T_1, ...T_m$. לפי ההגדרה של עץ m ארי כיוון שלכל תת עץ כזה חייב להיות לפחות עלה אחד, נקבל שכל אחד מהעלים מקיים את טענת האינדוקציה. נסמן את מס' העלים של כל עץ כזה ב־k_i ונקבל לפי הנחת האינדוקציה כי (פלוס אחד כי שורש העץ גם קודקוד פנימי)

$$\sum_{i=1}^{m} \frac{k_i - 1}{m-1} + 1 = \frac{k_1 + ... + k_m - m}{m-1} + 1 = \frac{n - m - (m-1)}{m-1} = \frac{n-1}{m-1}$$

כנדרש.

3202 סמסטר קיץ תרגיל בית 1 שאלה 5

שאלה: הצע מבנה נתונים דמוי מחסנית , מחסנית מינימום, אשר תומך בפעולות המחסנית הרגילות: אתחול, האם ריק, הכנסה לראש המחסנית,הוצאת איבר שבראש המחסנית והחזרתו למשתמש, וגם ־ החזרת איבר המינימום בכל רגע נתון (ללא הוצאה). הצע מבנה המשתמש במקום לינארי במס' האיברים במחסנית ומבצע כל פעולה בזמן קבוע. הראה שהמבנה אכן פועל בסיבוכיות המקום והזמן הנדרשות.

פתרון:

נשתמש בשני מחסניות. נסמנן s_1 ו־s_2. כל אחת מהמחסניות תהיה מחסנית רגילה.

המחסנית הראשונה תתנהל כמחסנית רגילה. אליה נדחוף את האיברים נוציא ונעשה כשנרצה. השנייה תהיה s_2 והיא תחזיק את איבר המינימום בכל רגע נתון.

0. אתחול ־ טריוואלי. כמו שראינו אתחול שני מחסניות ב־$O(1)$

א. הכנסה: בפעם הראשונה נכניס את האיבר הראשון לשני המחסניות. לאחר מכן ־ מהאיבר השניה והלאה נבצע בכל פעם בדיקה מול האיבר במחסנית s_2. כאשר נקבל איבר x נבדוק האם מתקיים $x < s_2.top$ ואם כן נעשה $s_2.push(x)$ וכמובן שאח"כ נעשה גם $s_1.push(x)$ ־ סה"כ תוארו כאן פעולות שהן ב־$O(1)$ ומדובר במס' סופי של פעולות ולכן סה"כ זמן קבוע.

ב. האם ריק ־ בדיוק כמו שעושים במחסנית רגילה, נשאל תמיד על s_1.

ג. הוצאת האיבר שבראש המחסנית ־ ראשית נחזיר כרגיל ונעשה $s_1.pop$ וזה אכן $O(1)$ ־ עכשיו צריך לבדוק שלא הוצאנו את המינימום. נבדוק ראשית לפני שנוציא האם $s_1.top == s_2.top$ ואם כן נבצע הוצאה של האיבר שבראש המחסנית s_2 גם כן ־ מאחוריו נמצא למטה איבר המינימום הקודם לפניו, ככה שאין בעיה. סה"כ גם כאן זמן קבוע של פעולות.

ד. החזרת איבר המינימום ־ פעולה אחת בדיוק $return - s_2.top$.
סיבוכיות מקום: השתמשנו בשתי מחסניות באורך עד n ולכן סה"כ תפסנו $2n$ מקום ־ כלומר $O(n)$ מקום ־ לינארי.

4202 תרגיל 2 שאלה 2

תהי קבוצה של מספרים $S = x_1, ...x_n$. נרצה מבנה נתונים שתומך ב: בנייה, שאילתת סכום ־ מחזירה האם קיימים שלושה איברים כך שמתקיים $x_i + x_k + x_j = y$. מצאו מבנה נתונים כזה שנבנה ב$O(nlogn)$ וזמן שאילתת סכום הוא $O(n^2)$.

פתרון: נקח את הקבוצה S ונמיין אותה, ראינו כבר שניתן למיין בדרכים רבות ב$nlogn$, למשל ע"י מיון ערימה. את המיון נשמור בתוך מערך A. כעת, אתחלנו את מבנה הנתונים שלנו.
עכשיו נרצה לבדוק האם קיימת שלשה כדקלמן.
נתבונן על האלגוריתם הבא.
נפתח לולאת for שתרוץ על כל איברי המערך. בכל איטרציה i נגדיר

$$x = arr[i]$$

$$k = arr[i+1]$$

$$m = arr[n-1]$$

נבדוק האם מתקיים $x + k + m = y$, אם כן נחזיר אמת וסבבה לנו.
אחרת,
*אם מתקיים כי $x + k + m > y$, נאמר כי $m = arr[lastIndex - 1]$ ונוריד את ערכו באחד
*אם מתקיים כי $x + k + m < y$ נגדיל את האינדקס של k באחד.
כך נעבור בוודאות על כל איברי המערך.
סיבוכיות: המקרה הגרוע ביותר הוא זה שאין שלישיה כזו. נבצע בסה"כ אכן $O(n^2)$ פעולות.

5202 תרגיל 2 שאלה 5

ידוע שהוכנסו למחסנית האיברים $1, ...n$ בזה אחר זה, והוצאו בסדר כלשהו. סדר ההוצאה של המספרים יוצר סדרה שהינה פרמוטציה כלשהי של המספרים הטבעיים עד n.
א. הצע אלגוריתם לבדיקה האם סידרה היא חוקית בזמן לינארי $O(n)$
ב. הצע אלגוריתם לבדיקה האם סידרה היא חוקית שסיבוכיות המקום הנוסף שלו היא $O(1)$
ג. עבור פלט q, נגדיר את qr כקריאה של q בסדר הפוך (כלומר, מימין לשמאל).
נאמר שסדרה היא עולה־יורדת אם קיים i עבורו $q[0], ...q[i]$ ממוינים בסדר עולה, ו־ .יורד בסדר ממוינים $q[i], q[i+1], ..., q[n-1]$
הוכח: מתקיים ש q ו־ qr חוקיות אם ורק אם היא עולה יורדת.
פתרון:
א. נתבונן באלגוריתם הבא ־
נשתמש במחסנית נוספת s_2ומשתנה $nextPush = 0$ שמייצג את המס' הבא שצריך להכנס למחסנית. כעת נעבור על כל S ∋s:
א. אם המחסנית אינה ריקה והמס' בראש המחסנית הוא s, כלומר יש להוציא את המספר. נבצע $s.pop$
ב. אם המחסנית ריקה או שs גדול מהמספר בראש המחסנית, צריך להכניס את כל המספרים מ$nextPush$ ועד לs למחסנית, נכניסם למחסנית ונעדכן $nextPush = s + 1$.
ג. אם s קטן מהמספר שבראש המחסנית, אזי זה מצב לא חוקי, כלומר לא יתכן שנכנס מס' גדול יותר לפניו. זוהי פרמוטציה לא חוקית ומיד נחזיר שגיאה.
סה"כ אם לא הגענו למצב לא חוקי בסעיף ג' נחזיר אמת. במקרה הכי גרוע נעבור על כל האיברים ונקבל $O(n)$ סה"כ זמן ריצה.

4202 תרגיל 4 שאלה 2

נתון מערך ממוין $A=[a_1,...a_n]$שבוצעו עליו k רוטציות. רוטציה היא פעולה שמוציאה את האיבר הראשון במערך ומכניסה אותו לסוף המערך. תארו אלגוריתם שמוצא את k.

פתרון:

יהא מערך A_kשבוצעו עליו k רוטציות.

א. אם מתקיים $A_k[n] > A_k[0]$ אזי נחזיר $k = 0$ כי זהו המערך המקורי ללא כל שינוי.

נבחר את האמצע של המערך. אם מתקיים $A[mid] < A[mid-1]$ אזי זה המקום שבו התחילה הרוטציה, כלומר $k = mid$.

אחרת,

אם $A[mid] > A[0]$ אזי השבירה לא התרחשה באזור זה, ונלך לחפש ב$mid+1 - -- > n$ כלומר בצד הימני.

אם $A[mid] < A[0]$ השבירה כן התרחש בצד הימני ונלך לאזור זה.

כך נפעיל את האלגוריתם שוב ושוב ונלך בכל פעם לאמצע. נראה כי מתקיים־

$$T(n) = 2T(\frac{n}{2}) + O(1) \Rightarrow O(logn)$$

3202 סמסטר קיץ תרגיל 2 שאלה ג3

נתון מערך A של מספרים חיוביים. נרצה לבנות מערך כך־

$$S[i] = max\{k|\forall_{i-k<j\leq i} A[j] \leq A[i] \wedge k \leq i\}$$

כלומר במיקום הi במערך S מופיע האורך של תת המערך הגדול ביותר $A[i-k,...i]$ שכל איברים קטנים מ$A[i]$.

א. כתוב אלגוריתם לבניית המערך S בסיבוכיות $O(n^2)$

פתרון: נרוץ על כל אינדקס $1 \leq i \leq n$ מה שיעלה לנו n פעולות. בכל פעם, נחפש את אורך המערך הגדול ביותר של איברים שקטנים מ$A[i]$. זה יהיה כבר טריוואלי ־ נעבור על המערך פעם אחת מה שיעלה לנו n פעולות: בכל פעם שנמצא איבר קטן מ$A[i]$ נעלה את אינדקס הסכימה שלנו ב1 (יאותחל בהתחלה לאפס). כך נמשיך עד שכבר יהיה לנו איבר שאינו קטן מ$A[i]$, כשנגיע אליו נשמור את האורך הכי גדול של מערך שמצאנו, ונאפס את אורך המערך וכך נמשיך. בכל פעם שנמצא אורך מערך רגע לפני האיפוס נבדוק אם הוא גדול מגודל המערך הגדול ביותר עד כה, ובמידת הצורך נעדכן בהתאם. זה אלגוריתם טריוואלי שעולה $O(n)$. כך נבצע n פעמים על כל אחד מאיברי המערך ונקבל סיבוכיות של $O(n^2)$.

ב. כתוב אלגוריתם משופר לבניית מערך זה ב$O(n)$.

פתרון: נרצה להשתמש הפעם במחסנית על מנת לייעל את זמן הריצה. נחזיק במחסנית זוגות את המידע אודות האיברים במערך: מיקום+ערך בזוגות, להעתיק לשם הכל יעלה לנו $O(n)$. נעבור על המערך משמאל לימין. בכל צעד i:

1202 תרגיל 2 שאלה 1

נתונה הפונקציה הבאה:

$$g : \mathbb{N} \rightarrow \mathbb{N}$$

המוגדרת ע"י

$$g(n) := \left\{ \begin{array}{cc} n & n = 1, 2, 3 \\ 5g(n-1) - 4g(n-3) & n \geq 4 \end{array} \right\}$$

תאר אלגוריתם שרץ בזמן ריצה $O(logn)$ שמחזיר את $g(n)$.

פתרון:

הפתרון הנאיבי שהוא להשתמש פשוט לפי הגדרת הרקורסיה יעלה לנו בצער רב $O(n)$. נרצה להשתמש בחזרות שמופיעות כאן שוב ושוב על מנת לשפר את זמן הריצה.
נעשה זאת ע"י שימוש במטריצה ווקטורים. נראה כי מתקיים לכל $n \geq 4$ כי

$$g(n) = 5g(n-1) + 0g(n-2) - 4g(n-3)$$

$$g(n-1) = 1g(n-1) + 0g(n-2) + 0g(n-3)$$

$$g(n-2) = 0g(n-1) + 1g(n-2) + 0g(n-3)$$

נראה כי

$$\begin{pmatrix} g(n) \\ g(n-1) \\ g(n-2) \end{pmatrix} = \begin{pmatrix} 5 & 0 & -4 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} * \begin{pmatrix} g(n-1) \\ g(n-2) \\ g(n-3) \end{pmatrix}$$

נסמן את המטריצה האמצעית ב־M. נראה כי מתקיים באופן כללי כי

$$\begin{pmatrix} g(n) \\ g(n-1) \\ g(n-2) \end{pmatrix} = \begin{pmatrix} 5 & 0 & -4 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^k * \begin{pmatrix} g(n-k) \\ g(n-k-1) \\ g(n-k-2) \end{pmatrix}$$

הk המקסימלי הינו $n-3$ ונראה כי לכל $n > 4$ מתקיים

$$\begin{pmatrix} g(n) \\ g(n-1) \\ g(n-2) \end{pmatrix} = \begin{pmatrix} 5 & 0 & -4 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{n-3} * \begin{pmatrix} g(3) \\ g(2) \\ g(1) \end{pmatrix}$$

$$\begin{pmatrix} g(n) \\ g(n-1) \\ g(n-2) \end{pmatrix} = \begin{pmatrix} 5 & 0 & -4 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}^{n-3} * \begin{pmatrix} 3 \\ 2 \\ 1 \end{pmatrix}$$

כלומר, על מנת להגיע למס' ה$g(n)$ נצטרך להעלות את המטריצה הנ"ל בחזקה קבועה, ראינו בתרגול כי זה עולה לנו $O(logn)$, וכן הכפלה בוקטור נוסף איננה משנה את היעילות של הפתרון, ואז כשנרצה לגשת לאיבר ה$g(n)$ זה יהיה האיבר בשורה הראשונה במכפלה.
סה"כ הסיבוכיות אכן $O(logn)$, כנדרש.

0202 תרגיל 7 שאלה 1

נתונה ערימת מינימום בינארית H עם n איברים שונים. כמו כן מס' טבעי $k \leq n$. הצע אלגוריתם אשר מחזיר את האיבר הk בגודלו בערימה בזמן $O(klogk)$.
פתרון: נשתמש בערימת עזר H_2, נבצע את התהליך הבא k שלבים: נעתיק מהערימה המקורית את איבר המינימום ונכניסו לתוך ערימת העזר. כמו כן נכניס לערימת העזר גם את שני בניו (בינארי). כעת מערימת העזר נוציא את איבר המינימום, אם בערימה המקורית יש לו ילדים נכניס אותם לערימת העזר. סה"כ בערימת העזר יהיה עד k איברים

ולכן עלות ההוצאה מערימת העזר היא $O(logk)$, כמו כן נבצע זאת k פעמים ולכן $klogk$. נשים לב שלאחר שנוציא k פעמים מערימת העזר, נקבל את האיבר הk בגודלו. נוכיח טענה זו באינדוקציה על k.

בסיס: $k = 1$, אם נרצה את האיבר ה1 בגודלו נעשה זאת ע"י החזרת האיבר הראשון ברשימה, נעתיקו לערימת העזר. הוא המינימלי ואותו נחזיר.

צעד: נניח נכונות עבור k. כעת נרצה את האיבר ה$k+1$ בגודלו להוציא, אנחנו יודעים כי לאחר k הוצאות קיבלנו את המינימלי וזה מהנחת האינדוקציה. כעת בראש הערימה המקורית עומד האיבר ה$k+1$ בגודלו, הוא נמצא גם בערימה H_2 כבן של האיבר הk בגודלו שהרגע הוצא, ולכן נמצא בראשה, ולכן אותו נחזיר. מש"ל.

5202 תרגיל 7 שאלה 2

נתון מערך A עם n איברים שונים שלמים. נתון כי A ממוין עם סיבוב k. כלומר מערך שלאחר הזזת כל האיברים k מקומות שמאלה באופן מעגלי נקבל מערך ממויין. הציעו אלגוריתם הפרד ומשול המוצא את האינדקס k.

פתרון: נשתמש בעקרון הפרד ומשול ונעבוד עם הערך האמצעי של המערך. נסמן $m = mid - index$. בכל פעם נתבונן על m ונבדוק:

א. אם $A[m] < A[m-1]$ נדע כי החל מאינדקס זה התחילו המעברים כלומר $k = mid$

ב. אם $A[m] > A[0]$ אזי המס' k נמצא היכן שהוא באזור $m+1 -- > n$ ולכן נלך לחפש שם.

ג. אם $A[m] < A[0]$ אזי המס' k נמצא היכן שהוא בצד השמאלי ונלך לחפש שם.

סה"כ סיבוכיות האלגוריתם תתואר כך

..

$$T(n) = T(\frac{n}{2}) + O(1) = O(logn)$$

2202 תרגיל 5 שאלה 2

במערך A של n מספרים ערך משמעותי הוא ערך המופיע יותר מ$\frac{n}{3}$ פעמים בA. כתבו אלגוריתם שמחזיר את כל הערכים המשמעותיים של A.

פתרון: נראה כי בהינתן נתוני השאלה יהיו במקסימום 2 ערכים משמעותיים. הפתרון הנאיבי הוא למיין את המערך, ואז לחפש האם קיימת סדרה של $\frac{n}{3}$ איברים זהים רציפים. זה יהיה לפחות $O(nlogn)$ ויש אפשרות ליותר טוב מזה. נרצה לעבור על האיברים פעם אחת ולדעת מיהם המשמעותיים. נעזר באלגוריתם סלקט שלמדנו בכיתה. נבצע את האלגוריתם על המערך A, מה שראינו שיעלה לנו $O(n)$, ונקבל מערך מסודר ככה ש$\frac{n}{2}$ האיברים הקטנים משמאל ו$\frac{n}{2}$ האיברים הגדולים מימין. נעשה זאת כעת על החצאים שוב, ונקבל שוב ארבעה תתי מערכים:

$$A_{LL}, A_{LM}, A_{MR}, A_{RR}$$

כך שלכל איבר ברבע השמאלי קטן שווה מכל איבר ברבע A_{LM}שקטן שווה מכל איבר ברבע A_{MR} שקטן שווה מכל איבר ברבע הימני.

ערך משמעותי חייב להמצא ביותר מ$\frac{n}{3}$ מהמקומות במערך. כמו כן, ברור כי $\frac{n}{3} > \frac{n}{4}$. ולכן אנחנו נבצע כך :

ייתכן כי ערך משמעותי יחל באמצע רבע אחד ויסתיים ברבע העוקב לו. מה לא ייתכן? שלא יהיה ערך משמעותי בחציונים. כלומר - אחד מן החציונים, או של A (נסמנו x_A) או של מחצית המערך הימני (נסמנו X_R) או של מחצית המערך השמאלי (X_L) הוא חציון. כעת נשמור את כל האיברים הללו במערך.

נעבור פעם נוספת ואחרונה על כל האיברים, כאשר בכל פעם נבדוק שלושה דברים: אם האיבר שלנו שווה לאחד מx-ים, אם כן נעלה את הקאונטר שלהם באחד. לבסוף, נעבור על שלושת הקאונטרים של שלושת האיברים, ואם יש איבר שערך הקאונטר שלו גדול מ$\frac{n}{3}$, נחזירו והוא ערך משמעותי.

סיבוכיות: בזכות אלגוריתם סלקט ביצענו $O(n)$ פעולות, פלוס מעבר שלנו על המערך פעם נוספת בסוף וכן

$$O(n) + n = O(n)$$

הערה: נוכיח פורמלית שאם וכאשר יש איבר משמעותי, הוא אכן אחד מהחציונים.

יהי x איבר משמעותי, נב"ש כי איננו חציון, ואיננו חציון של חצאי המערכים. x צריך לקיים כי הוא חוזר על עצמו $\frac{n}{3}$ פעמים. מכאן, אם x יופיע בשליש הראשון של המערך, הוא בוודאות יהיה בחציון של החלק השמאלי. בסתירה.

אם x יופיע בשליש השני של המערך, הוא בוודאות יהיה בחציון של המערך כולו, בסתירה. אם x יהיה בשליש האחרון של המערך הוא יהיה בחציון החלק הימני של המערך, בסתירה. אם x מתחיל בשליש הראשון ומסתיים בשני, שוב, הוא יתקל בוודאות בחציון המערך. ואם מתחיל בחצי השני ומסתיים בשלישי, יתקל בחציון של החלק השמאלי. לא קיימות אפשריות אחרות. ולכן הטענה הוכחה.

שאלה 3 תרגיל 5 3202 קיץ

הבעיה: נתונות שתי רשימות L_1 ו-L_2. כתוב אלגוריתם שרץ בזמן הריצה הטוב ביותר בממוצע שבודק האם האיברים בשתי הרשימות זהים (לאו דווקא באותו סדר, אין חזרות)

פתרון:

1. נכניס את כל איבר L_1 לטבלת האש אחת. ראשית, אם $|L_1| \neq |L_2|$ אזי בוודאות נחזיר שקר ־ כי אין חזרות ולכן בוודאות קיים איבר ברשימה אחת שלא נמצא בשנייה. אחרת נכניסם לטבלת האש עם פונקציית גיבוב כמעט מושלמת. זה יעלה לנו $O(n)$. עבור כל איבר ברשימה L_2, נשלוף את ערכו ונחפש האם ערך זה נמצא בטבלת האש. חיפוש בטבלת האש עולה $O(1)$ ולכן סה"כ אנחנו נרצה לעבור במקסימום על n חיפושים כאלו, אם אכן בכל אחד מהחיפושים מצאנו איבר אזי נחזיר אמת. נשים לב שבכל פעם שנמצא ערך מתאים בתוך טבלת האש, נסירו מהטבלה $O(1)$. כעת, כשנגיע לסוף אם טבלת האש שלנו תהיה ריקה ללא איברים, אזי בוודאות הרשימות זהות. שכן אין חזרות.

זמן הריצה שלנו: בנינו טבלת האש $O(n)$, חיפשנו בטבלת האש אחת n, סה"כ $O(n)$.

שאלה 4 תרגיל 5 3202 קיץ

הבעיה: יש קלט של n מספרים הלקוחים בהתפלגות שווה מהקטע $(0,1)$. עלינו לבדוק האם ישנם שני מספרים המקיימים $a^2 = b$. כתוב אלגוריתם יעיל לפתרון בעיה זו 1. במקרה הגרוע, 2. במקרה הממוצע.

פתרון: למקרה הגרוע, לא נוכל להשתמש בטבלת האש לצערנו. נכניס את הנתונים למערך ונמיין איכשהו ב-$O(nlogn)$. כעת, עבור כל קלט x נבדוק האם הערך x^2 נמצא במערך. נעשה זאת באמצעות חיפוש בינארי (בקפיצות של 2) שיעלה לנו בכל פעם $logn$. לכן סהכ החיפוש בפנים למקרה הכ גרוע יהיה $nlogn$ ונקבל $2nlogn = O(nlogn)$.

למקרה הממוצע, נשתמש בטבלת האש. נניח כבר שהם כן נבחרו בהתפלגות שווה. נכניס את כל המספרים לטבלת האש. זה יעלה לנו $O(n)$. עבור כל מספר, נחפש האם x^2 קיים, כזכור חיפוש בתוך טבלת האש עולה $O(1)$. כך נעשה עבור n המספרים ונקבל סה"כ סיבוכיות של

$$O(n) + nO(1) = O(n)$$

רביעייה פתגורית

קלט: מערך A המכיל מספרים שלמים וחיוביים.

פלט: האם קיימים a,b,c,d כך ש-$d = \sqrt{a^2+b^2+c^2}$

דרישות: $O(n^2)$ בממוצע או $O(n^2logn)$ במקרה הגרוע.

פתרון:

נשים לב שמשיקולים מתמטיים, הדבר שקול לבדוק האם קיים $d^2 = a^2+b^2+c^2$.

נעבור על כל האיברים במערך ובכל איבר נעלה את ערכו בריבוע. מעבר על n איברים עם פעולות קבועות יעלה לנו $O(n)$. כעת נסמן את המערך עם הערכים החדשים כ-A'. לכל איבר מתקיים כי

$$A'[i] = (A[i])^2$$

ולכן הדבר כעת שקול לבדוק האם קיימים ארבעה מספרים שמקיימים

$$d' = a' + b' + c'$$

הדבר שקול ל

$$d' - a' = b' + c'$$

כמה זוגות אפשריים יש לסכומים בכל אחד מהצדדים? n^2זוגות. לכן נכניס את כל הזוגות הללו לטבלת האש, כעת נעבור על כל n^2 האפשרויות של $d' - a'$ ונחפש עבור כל אפשרות האם קיים מישהו ששווה להם בטבלת האש, אם מצאנו אחד כזה אזי סיימנו את הבעיה. נזכיר שחיפוש בתוך טבלת האש הוא $O(1)$. מכאן שסה"כ נבצע n^2 חיפושים.
סה"כ ביצענו n פעולות בהתחלה, אח"כ n^2 חיפושים, n^2 העלות הכנסה לתוך טבלת האש, ולכן הסיבוכיות שלנו היא $2n^2 + n = O(n^2)$.

תרגיל 3 שאלה 3 4202

הציעו מבנה נתונים התומך בפעולות הבאות:

- הכנסה
- מחיקה
- עוקב k ־ זא: אם היינו מחזיקים את הערכים במערך ממוין, אז בהינתן ערך x הנמצא במבנה, ומספר k, אנו רוצים להחזיר את האיבר שנמצא k מקומות מימין ל־x במערך הממוין.

יש לתאר מבנה נתונים בו כל אחת מהפעולות מתבצעת בזמן $O(logn)$ כאשר n הוא מספר האיברים במבנה

פתרון:

נשתמש בעץ AVL. הכנסה ומחיקה מהעץ כפי שראינו בכיתה היא $O(logn)$. כעת צריך לדבר על פעולת עוקב k. אנחנו יודעים שבעץ AVL הוא עץ חיפוש, כלומר האיברים ממויינים. בהינתן מס' X ניתן לחפש אותו בעץ בעלות של $O(logn)$. נחפש אותו, כעת, אנחנו נרצה לחפש את האיבר שנמצא k מקומות ימינה ממנו. לשם כך אנחנו נשמור בתוך העץ מידע מורחב. ראינו את הטריק הזה כבר בעבר. נוסיף לכל איבר ערך $size$ שיגיד כמה איברים יש בתת העץ שלו. זה לא ישנה לנו את הסיבוכיות. כמו כן גם לעדכן במחיקה והכנסה זה לא ישנה יותר ממס' פעולות קבוע. מדוע זה עוזר לנו להוסיף את השדה $size$? בהינתן שאנחנו יודעים את מס' האיברים בעץ הנוכחי, אנחנו נדע היכן לחפש את האינדקס k. כלומר, אם $k > size$, בוודאות נצטרך ללכת לתת העץ שמימיני. אם $k < size$ אזי זה בתוך תת העץ שלי מימין. אנחנו עושים מס' פעולות חיפוש קבועות בתוך עץ קטן יותר, ולכן סה"כ הסיבוכיות היא $O(logn)$.

תרגיל המטריצות מהאתר של גלעד

שאלה 1. (ממבחן + תוספת) מספרי פיבונאצ'י מוגדרים ע"י

$$F_i = F_{i-1} + F_{i-2}, \qquad F_0 = 0, F_1 = 1$$

נרצה לחשב את F_n מבלי להשתמש בנוסחא מפורשת, אלא בעזרת נוסחא רקורסיבית. כפי שכבר ראינו בתרגיל קודם - פתרון התוכנית הרקורסיבית בצורה הנאיבית - יקרה. נרצה לעשות זאת מהר יותר. לכל האורך, נניח שכל פעולת כפל, וכמו-כן - כפל של שתי מטריצות מגודל 2×2 עולה $O(1)$ זמן.

1. תהי $A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$. נשים לב שמתקיים: $\begin{pmatrix} F_n \\ F_{n-1} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} F_{n-1} \\ F_{n-2} \end{pmatrix}$.
   הסק מכאן נוסחא ל - $\begin{pmatrix} F_n \\ F_{n-1} \end{pmatrix}$ כפונקציה של-A, ושל $\begin{pmatrix} F_1 \\ F_0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$.
2. כאמור, נרצה לחשב אם כן את A^k בזמן פחות מ - $O(k)$. נניח וחישבנו את A^8. האם צריך עוד - 8 פעולות של כפל מטריצות בכדי לחשב את A^{16}? הראה/י כיצד לעשות כן בפעולה אחת.
3. לפי אותו עיקרון כמו בסעיף הקודם, כמה פעולות נדרשות בכדי לחשב את A^8. הכלל לכל k, כאשר k חזקה שלמה של 2 (כלומר, הצג אלגוריתם). בנוסף, הצג נוסחא רקורסיבית לזמן הריצה, ופתור אותה (בעזרת אחת השיטות שלמדנו).
4. איך נחשב את A^5? ואת A^{12}? הכלל לחזקה k כלשהי, לא בהכרח חזקה שלמה של 2.
5. סיכום - בהינתן k - כמה עולה האלגוריתם שהצגת לחישוב F_k?
6. שיפור של האלגוריתם הקודם. נשים לב כי לכל x, ולכל $n > 0$ שלם מתקיים:

$$x^n = \begin{cases} 1 & \text{if } n = 0 \\ x \cdot \left(x^{\frac{n-1}{2}}\right)^2 & \text{if } n \text{ is odd} \\ \left(x^{\frac{n}{2}}\right)^2 & \text{if } n \text{ is even} \end{cases}$$

   כתוב אלגוריתם (רקורסיבי) המקבל x ומחזיר x^n. חשב סיבוכיות זמן ריצה של האלגוריתם (בעזרת נוסחת נסיגה. אפשר וצריך "לעגל פינות").

פתרון:
א.

$$\begin{pmatrix} f_{n+1} \\ f_n \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} f_n \\ f_{n-1} \end{pmatrix}$$

מדובר בנוסחה רקורסיבית עם וקטור! נראה כי באופן כללי מתקיים

$$\begin{pmatrix} f_{n+1} \\ f_n \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^i \begin{pmatrix} f_{n+1-i} \\ f_{n-i} \end{pmatrix}$$

כלומר מתקיים כי עבור $i = n$

$$\begin{pmatrix} f_{n+1} \\ f_n \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

ב. נניח ובידינו A^8. נראה כי מתקיים

$$A^{16} = (A^8) * (A^8)$$

מכאן שנוכל לעשות כפל של שתי מטריצות בלבד! כאשר את A^8כבר יש לנו. מכאן שזה לא יעלה 8 פעולות אלא כפל אחד בלבד ־ במונחי יעילות זה $logn$.
ג. נניח כי K הינו חזקה של 2. כלומר קיים $n \in \mathbb{N}$ כך ש$2^n = k$. כעת, אנחנו נרצה למצוא פתרון רקורסיבי שפותר את הבעיה לכל $k \in \mathbb{N}$. ובכן ־ כיוון שk זוגי, כיוון שהינו חזקה של 2, וכן נניח כי הוא שונה מ1 ($2^0 = 1$), שכן אם הוא אחד אז אין מה לחשב, אזי ־ תמיד יתקיים

$$A^k := A^{\frac{k}{2}} * A^{\frac{k}{2}}$$

כעת, בשביל לייצג את הבעיה באמצעות נוסחת נסיגה נשים לב. אנחנו בכל פעם נחשב את המטריצה בחזקת חצי מקיי פעם אחת, ואז אנחנו נכפול בעצמה, זה מס' קבוע של פעמים! מכאן שנוסחת הנסיגה תהיה־

$$T(n) = T(\frac{n}{2}) + O(1)$$

ובכן

$$n^{log_2 1} = n^0 = 1 = O(1)$$

ולכן

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

כנדרש.
ד. כיצד נחשב את A^5למשל? עלינו להכליל את הפתרון לחזקה כלשהי k.
נתבונן באלגוריתם הבא:

$$A^k := \left\{ \begin{array}{cc} A & k=1 \\ A^{\frac{k}{2}} * A^{\frac{k}{2}} & 2|k \\ A^{\frac{k+1}{2}} * A^{\frac{k-1}{2}} & o.w \end{array} \right\}$$

נשים לב שבהינתן שאכן נגיע ל$o.w$ אזי k הינו אי זוגי, אם נוסיף אחד או נוריד אחד נקבל מס' זוגי ואז הפתרון יוכל ללכת לפי המצב בו k זוגי. סה"כ נוסחת הנסיגה גם כאן תתן כמו קודם, $O(logn)$
ה. ראינו בסעיף א' כי מתקיים

$$\begin{pmatrix} f_{n+1} \\ f_n \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

ראינו קודם כי לחשב את המטריצה בחזקת n יעלה לנו $O(logn)$ וכן להכפיל בוקטור זה קבוע, ואז אנחנו נקבל וקטור מהתוצאה. בשביל למצוא את f_nנלך לאיבר שנמצא בוקטור בשורה השנייה, זו גישה ב$O(1)$. סה"כ למצוא את f_nיעלה לנו $logn$ פעולות. כלומר $O(logn)$.

שאלה 3 ־ מטלה 5 קובץ גלעד

הבעיה: בהינתן מערך ממוין A של n מספרים שלמים שונים, כאשר יתכן שחלקים שליליים, תאר אלגוריתם המוצא אינדקס $1 \leq i \leq n$ i כך ש$A[i] = i$. אם לא קיים כזה, החזר לא קיים. אם קיימים כמה כאלו, החזר אחד כלשהו כזה.
פתרון:
הפתרון הנאיבי יהיה מעבר על כל המערך, עבור כל איבר נבדוק האם $A[i] = i$, ואם מצאנו נחזיר אחרת לא. זה יעלה לנו $O(n)$ וברור כי זו לא הייתה כוונת המשורר.
כלומר נצטרך לפתור את זה ב$O(logn)$. המערך ממוין וזה עוזר לנו. אנחנו נלך בכל פעם לאמצע המערך, אנחנו נחשב את הערך ונבדוק האם $A[mid] = mid$. אם אכן כך מעולה כי מצאנו איבר ונחזיר, אם $A[mid] > mid$, אנחנו

יודעים כי כל האיברים מעתה ואילך לא יהיו רלוונטים שכן כל המספרים הינם שונים ולכן לא יתכן שיופיע מס' אחריו שמקיים $A[mid] = mid$, לכן נפעיל את הבעיה על $[0, ..mid-1]$. אם $A[mid] < mid$, לא יתכן כי יופיע איבר כזה לפניו. לכן נחפש ב$[mid+1, ...n]$ מכאן שנקבל

$$T(n) = T(\frac{n}{2}) + O(1) = O(logn)$$

שאלה 6 - מטלה 5 קובץ גלעד

הבעיה: נתון מערך בן n קומות. ונתונים k כדורי בדולח. כולם זהים. אנו מניחים כי קיימת קומה ℓ כך ש- $1 \leq \ell \leq n$ כלשהי כך שאם נזרוק את הכדורים בקומה נמוכה מ-ℓ הכדורים ישארו שלמים, אך החל מהקומה ℓ הכדורים מתנפצים. כמובן שלא ינתן להשתמש בכדור עזיר לאחר שהתנפץ. אנחנו נדרשים לתאר אלגוריתם ולמנות את זמן הריצה שלו, כאשר המטרה היא למצוא את הקומה ℓ עם מספר מינימלי של זריקות, כאשר:

$(1)k = 1(2)k = 2(3)k = logn$

פתרון:

נתחיל עבור $k = 1$. אין ברירה בצורה הנאיבית אלא לעבור קומה קומה כאשר נתחיל מלמטה למעלה ובוודאות נמצא קומה זו. זה $O(n)$

עבור $k = 2$, כעת יש לנו כדור עזר שיוכל לעזור לנו. נחפש ב$\sqrt{n}$קומות, כאשר בכל פעם נבצע קפיצה לקומה הבאה, אם נשבר הראשון אזי יש לנו טווח של $\sqrt{n}$לחפש בו, לכן סה"כ $O(n^{0.5})$

עבור $K = logn$, כעת חיינו קלים יותר. נוכל לזרוק מהאמצע, אם נשבר נחפש בקומות שקטנות מהאמצע ואם לא אז באלו שגדולות מהאמצע, בסוף לאחר $logn$ כדורים אכן נמצא את הקומה המנייאקית. סה"כ $T(n) = T(\frac{n}{2}) + O(1) =$ $O(logn)$.

מטלה 9 קובץ גלעד

א. הבעיה: בהינתן שני מערכים ממוינים m וn הראה אלגוריתם שפועל ב$O(m+n)$ ומחזיר מערך חדש ממויין.

פתרון: נחזיק מצביעים לאיבר הראשון בכל מערך, בכל איטרציה נבדוק מי מהאיברים יותר קטן ואותו נכניס לחדש, ונזיז כך את המצביע.

ב. הבעיה: בהינתן k מערכים שמס' האיברים הכולל בהם הינו n, כך שכל אחד מהמערכים ממויין, כתוב אלגוריתם שמראה כיצד ניתן למזג את כל המערכים בזמן $O(nlogk)$

פתרון: בדומה לסעיף הקודם, אנחנו הולכים הפעם להשתמש בערימה. נקח את האיבר הראשון בכל מערך, יש סה"כ k איברים כאלו ונבנה מהם ערימה. נכניס את כל האיברים לערימה, לבנות ערימה עולה $O(k)$, ואז נשלוף את ראש הערימה - שהרי אנו בערימת מינימום. אותו נכניס למערך, בכל פעם לכל איטרציה, נוציא את המינימלי ביותר - ראש העץ, ונכניס את האיבר הבא באותו מערך בדיוק אחריו, כלומר קצת יותר פורמלי: בהינתן איבר a_{ij} בתוך מערך $1 \leq j \leq k$, אזי בהינתן שa_{ij} היה המינימלי בראש הערימה, נוציא אותו, ונכניס לערימה את a_{i+1j}. להכניס ולהוציא מערימה - שתי פעולות קבועות, ב$O(logk)$. כמה פעמים נבצע את ה"for" הזה? בדיוק n פעמים, שכן יש סה"כ n איברים. סה"כ נבצע n פעמים $logk$ ונקבל סיבוכיות של $O(nlogk)$

ג. הבעיה: נתונה רשימה של n מספרים. ברצוננו לבצע את הלולאה הבאה $n-1$ פעמים: הוצא את שני המספרים הקטנים ביותר, x ו-y, מהרשימה, והדפס אותם. צור מספר חדש שהוא סכום המספרים שהוצאו, ז"א $z = x + y$. הכנס את z לרשימה.

אלגוריתם פשוט לביצוע המשימה ייקח זמן $O(n^2)$, אפילו אם המספרים נתונים כבר בצורה ממויינת. תאר/י מבני נתונים ואלגוריתם מתאים שיבצעו זאת בזמן $O(nlogn)$.

פתרון: נשתמש בערימת מינימום כמובן, ניצור אותה ב$O(n)$ ונוציא את האיבר המינימלי, זה יעלה $O(logn)$. אח"כ נוציא את ההבא המינימלי. גם זה יעלה $O(logn)$. נשמור לנו אותם ונגדיר $z = x + y$. נכניס אותו לערימה, גם זה עולה $O(logn)$. סה"כ ביצענו כאן $3logn$ פעולות. נצטרך לעשות זאת $n-1$ פעמים, ונקבל

$$3logn(n-1) = 3nlogn - 3logn < 3nlogn = O(nlogn)$$

ד. בשני הסעיפים הבאים מותר להניח כי n, m הם חזקות של 2 לשם הפשטות.

1. נתונים שני מערכים ממויינים בגודל n. הראו אלגוריתם המוצא את החציון של איברי שני המערכים בזמן $O(logn)$.

2. נתונים שני מערכים ממויינים, האחד בגודל n, השני בגודל m. הראו אלגוריתם המוצא את החציון של איברי שני המערכים בזמן $O(log(n+m))$.
פתרון:
עבור הבעיה הראשונה: אנחנו הולכים לרוץ במקביל על שני המערכים. כיוון ששניהם ממוינים נוכל לעשות זאת באופן הבא: נסמן את המערכים A, B ונרוץ כך: בכל איטרציה נלך לאיבר האמצעי ביותר $a_{\frac{n}{2}}$. אם האיבר האמצעי של A קטן מהאיבר האמצעי של B, נתעלם לחלוטין מהצד השמאלי של A ומהימני של B ואם האיבר האמצעי של A גדול מהאיבר האמצעי של B אזי נתעלם מהצד הימני של A ומהשמאלי של B. סה"כ בסוף אנחנו נמצא את החציון כי נקבל מערך בגודל 1, נראה כי

$$T(n) = T(\frac{n}{2}) + O(1) = O(logn)$$

עבור הבעיה השנייה: הפעם נרצה להבין כיצד לחתוך את שני המערכים מהצדדים כמו קודם, אך לקבל חלוקה זהה. שכן $m \neq n$ (אם הם שווים הפתרון מעלה עובד). נניח $m > n$ בלי הגבלת הכלליות. נרצה למצוא חלוקה נכונה, כזו שכל האיברים בקבוצה השמאלית יהיו קטנים שווים מאלו בימנית בתוך המערך. נרצה בכל פעם להשאר עם $\frac{m+n}{2}$ האיברים הטובים לנו, ואז לקבל נוסחת נסיגה כזו:

$$T(m+n) = T(\frac{m+n}{2}) + O(1) = O(log(m+n))$$

ואז לכאורה סיימנו, השאלה היא מיהם ה$\frac{m+n}{2}$ הטובים לנו. אנחנו הולכים לרוץ על i ולבדוק נסיונות חתך. כלומר לכל $1 \leq i \leq \frac{m+n}{2}$ אנחנו נגדיר $j = \frac{m+n}{2} - i$ על מערך B ונקח i איברים מהסוף של A וj איברים מהסוף של B. בכל פעם כזו נצטרך לבדוק האם $A[i-1] \leq B[j]$ כלומר אם האיבר הימני ביותר של B גדול מהאיבר השמאלי ביותר של A וכן אם $B[j-1] \leq A[i]$. אם כן אזי החציון בסוף יהיה $max\{A[i-1], B[j-1]\}$, אם החלוקה לא נכונה, נסדר עם חיפוש בינארי .

שאלה 3 - מטלה 3 - קובץ גלעד

בתרגול למדנו מערך דינמי הגדל פי שניים בכל פעם שאין מקום, התבוננו בפעולת ההכנסה בלבד. בשאלה הבאה נתייחס למערך מלא בגודל n. נניח כי במצב ההתחלתי הוא מלא לגמרי. אנו מתבוננים רק בפעולת הוצאה של האיבר האחרון מהמערך, לאחר רצף של פעולות, כאשר במערך נותרים $\frac{n}{2}$ איברים, מקצים מערך חדש בגודל $\frac{n}{2}$, מעתיקים את האיברים למערך החדש ומשחררים את המערך הישן. לסעיפים הבאים השתמש בפונקציית הפוטנציאל הבאה:

$$\phi(D_i) = size - num$$

כאשר $size$ הוא הגודל הנוכחי של המערך, וnum הוא מס' האיברים בפועל של המערך.
א. הוכח: לכל $i < n$ $\phi(D_i) \geq \phi(D_0)$. הנח לשם כך שבמצב ההתחלתי D_0 גודל המערך הינו n ויש בו בדיוק n איברים. הסק, שלכל $m < n$ מתקיים

$$\sum_{i=1}^{m} \hat{c_i} \geq \sum_{i=1}^{m} c_i$$

ב. הנח סדרה של n פעולות הוצאה. הצע חסם לסדרת פעולות זו, באמצעות שיטת הפוטנציאל.
פתרון:
א. נראה כי במצב ההתחלתי מתקיים $size = num$ ולכן $\phi(D_0) = size - num = 0$. כלומר, ההוכחה שלנו הפכה להיות $\phi(D_i) \geq 0$ באופן שקול לכל $i < n$. נוכיח טענה זו כעת. נראה כי יש לנו פעולה אחת בדיוק $\sigma_i \in$ {הוצאה של איבר}, נב"ש כ $\phi(D_i) < 0$. אזי, קיים שלב i כך ש$size - num < 0$, כלומר $size < num$. אם כן, אי השוויון כאן אומר כי יש כרגע יותר איברים במערך מאשר גודלו! זה בסתירה לכך שהפעולה היחידה שנתמכת היא פעולה ההוצאה, ולכן $\phi(D_i) \geq 0$ לכל $i < n$.
כעת נשים לב כי

$$\sum_{i=1}^{m}\hat{c_i}=\sum_{i=1}^{m}c_i+\phi(D_i)-\phi(D_{i-1})=\sum_{i=1}^{m}c_i+\phi(D_n)-\phi(D_{n-1})+\phi(D_{n-1})-\phi(D_{n-2})+......-\phi(D_0)=$$

$$\sum_{i=1}^{m}c_i+\phi(D_n)-\phi(D_0)\geq\sum_{i=1}^{m}c_i$$

ממה שהוכחנו בתחילת הסעיף, כנדרש.
ב. נתבונן על סדרה של n פעולות הוצאה. נחלק למקרים ־ אם הפעולה הi לא גורמת להקטנת מבנה הנתונים:

$$size_{i-1}=size_i,num_i=num_{i-1}-1,c_i=1$$

$$\phi(D_i)-\phi(D_{i-1})=size_i-num_i-(size_{i-1}-num_{i-1})=$$

$$size_i-size_{i-1}+num_{i-1}-num_i=0+1=1$$

ולכן

$$c_i+\phi(D_i)-\phi(D_{i-1})=2$$

אם הפעולה הi כן גורמת להקטנת מבנה הנתונים, אזי

$$size_{i-1}=2size_i,num_{i-1}=num_i,c_i=size_{i-1}=2size_i=num_i=num_{i-1}$$

$$c_i+\phi(D_i)-\phi(D_{i-1})=2size_i+size_i-num_i-(size_{i-1}-num_{i-1})=$$

$$2size_i+size_i-size_{i-1}+num_{i-1}-num_i=0$$

סה"כ לכל סוג פעולה, עבור סדרה של n פעולות נהיה חסומים לפי הסעיף הקודם והנוכחי ב$O(2n)$. לשיעורין לכל פעולה, 2.

5202 תרגיל בית 5 שאלה 4:

מחסנית עם ביטול: נניח שיש לך מבנה נתונים של מחסנית שתומך בפעולות פופ פוש ו$UNDO$: החזר את המחסנית למצב בו היא הייתה לפני k הפעולות האחרונות. כלומר מבצעים חזרה לאחור של k הפעולות.
הניתוח השיעורי מתבסס על כך שכל פעולה דורשת $O(1)$ זמן אך פעולת הביטול יכולה להיות מורכבת . בצע ניתוח לשיעורין על פעולת הביטול. שאלות עזר ־ מהי העלות של פעולת הדחיפה וההוצאה הרגילות? כיצד ניתן לנתח את

העלות הממוצעת של פעולת הביטול? הוכח שהעלות הממוצעת של כל פעולה כולל הביטול היא $O(1)$,אם אם על פניו נראה כי היא $O(k)$.

פתרון: ננסה לנתח מה עלינו לעשות. על פניו, נרצה לשמור מחסנית נוספת שבה יהיו את הפקודות שלנו. על פניו אם נפעיל את פעולת הביטול, אנחנו נמשוך k פעמים מהמחסנית של הפעולות את הפעולות ושנחזרן על המחסנית שלנו, כל אחת מהן עולה $O(1)$, ולכן סה"כ פעולת ביטול תהיה $O(k)$. האם כך הדבר? ננתח זאת שונה. נפעול לפי שיטת המטבעות - בכל פעולת הכנסה ניתן שני מטבעות, וכך גם בכל פעולת הוצאה. כעת ננתח כל אחת מהפעולות שלנו:

הכנסה - נכניס 2 מטבעות. באחת נשתמש באופן אוטומטי, ואחת נשלח לבנק. הוצאה - גם כן 2 מטבעות, באחת נשתמש באופן אוטומטי, ואחת נשלח לבנק. בעצם המטבע ש"ישלח" יהיה זה שישלם על הקריאה העתידית של הפעולה ממחסנית הפעולות. כעת שניגש לפעולת הביטול נראה כי - יש כרגע בבנק שלי k מטבעות לפחות, מדוע k? כיוון שעד כה בוצעו k פעולות $push$ או pop ועל כל אחת מהם שמרנו מטבע אחד. אנחנו נשתמש בכל המטבעות מהבנק, כלומר ניקח את כל k המטבעות ולכן הניתוח לשיעורין יהיה:

$$\hat{c_i} = c_i + deposit - withdraw$$

כמה העלות האמיתית? k, כמה מפקידים? 0. כמה יש בבנק? k ולכן

$$\hat{c_i} = k + 0 - k = 0$$

עבור פעולת $push$: עלות הפעולה היא 1, בנוסף נכניס אחד לבנק ולכן:

$$\hat{c_i} = 1 + 1 - 0 = 2$$

עבור פעולה pop: עלות הפעולה היא 1, בנוסף נכניס אחד לבנק ולכן:

$$\hat{c_i} = 1 + 1 - 0 = 2$$

סה"כ לסדרה של n פעולות $T(n) \leq 2n$ ולכן העלות הממוצעת עבור פעולה היא 2. בפרט $O(1)$ לכל סוג פעולה.

2023 קיץ מועד א' אילת

שאלה 1:

נאמר כי מערך בגודל n הוא מערך בצמיחה או"א אם:

א. $n = 1$.

ב. אחרת, לכל $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$ ולכל $\lfloor \frac{n}{2} \rfloor + 1 \leq j \leq n$ מתקיים $A[i] < A[j]$ וגם $A[\lfloor \frac{n}{2} \rfloor + 1, ...n]$ מערך בצמיחה.

הוכח הפרך: לכל מערך A בגודל n קיים אלגוריתם שמקבל את המערך A ומחזיר את המערך B שהוא מערך בצמיחה בגודל n וערכי המערכים זהים (מיקומים לא בהכרח) בזמן $O(n)$

פתרון:

הוכחה: נשתמש באלגוריתם סלקט על האיבר ה$\lfloor \frac{n}{2} \rfloor$ בגודלו. כפי שלמדנו לכל איבר $\lfloor \frac{n}{2} \rfloor + 1 \leq j$ וכן $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$ יתקיים $A[i] < A[j]$ כי האלגוריתם לוקח את המערך ושם לאחר איבר ה$pivpt$ את אלו שגדולים ממנו מימין וקטנים ממנו משמאל.

כעת, קיבלנו את התנאי הראשון. כעת נלך להסתכל על תתא המערך $\lfloor \frac{n}{2} \rfloor + 1, ...n$. נחזור על אותה פעולה - נבצע סלקט על האיבר החצי בגודלו כאן. זה יעלה $\frac{n}{2}$ על גודל הקלט. לאחר מכן שוב נקבל כמו קודם, מסודרים מימין ומשמאל. כך נמשיך עם האלגוריתם עד שנגיע לתא בגודל 1 ושם נפסיק וקיבלנו מערך שעונה על דרישותנו. סיבוכיות -

$$n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + ... = \sum_{k=1}^{log_2 n} \frac{n}{2^k} \leq \sum_{k=1}^{\infty} \frac{n}{2^k} = 2n = O(n)$$

שאלה 2:

סעיף א - הציעו מבנה נתונים S שתומך בפעולות הבאות

א. $push(S,x)$ הכנסת האיבר x לS. $x \in \mathbb{R}$.

ב. $POP(S)$ מחיקת האיבר האחרון שנכנס לS

ג. $Reduction(S,d)$ הורדת הערך הממשי d לכל איברי המבנה S

ד. החזרת ערך מקסימלי במבנה S

עלות כל אחת מהפעולות $O(1)$

סעיף ב - הוכח הפרך: ניתן להוסיף למבנה הנתונים פעולת מחיקת מקסימום ב$O(1)$ זמן.

פתרון:

נעבוד עם מחסנית כמובן. מסגרו לנו שזה המבנה הנדרש. נראה כי הכנסה ומחיקה זה לא דבר קשה במיוחד - זה פוש ופופ קלאסי ב$O(1)$. גם למצוא את הערך המקסימלי בכל רגע נתון זה לא קשה - נעבוד עם מחסנית מקסימומים בכל רגע נתון נגדיר $max = -\infty$ בהתחלה ונבדוק האם האיבר הראשון גדול ממנו נגדירו כמקסימום וכך בכל כניסה של איבר נכניס מקסימום חדש. כשאיבר יעזוב נמחק איבר מראש מחסנית מקסימומים. סה"כ זה יהיה למימוש במספר קבוע של פעולות $O(1)$. כלומר - לא לשם כך התכנסנו.

הבעיה המעניינת יותר היא סעיף ג'. נרצה להוריד את הערך הממשי d לכל איברי המבנה. ובכן, לא באמת ניתן לעשות זאת ב$O(1)$. הרי חייבים לעבור על כל איברי המבנה. זה לפחות $O(n)$ אם נרצה לעדכון מיידי. לכן הגישה תהיה כזו - המשתמש לא יודע מה קורה בתוך המחסנית שלי בכל רגע נתון, אלא רק שהוא מקבל איבר אליו חזרה למשל. אנחנו הולכים לנהל משתנה בשם x שיהיה קאונטר וכן הולכים להוסיף שדה לכל איבר בעת כניסתו למבנה. השדה שנוסיף יקרא $sumWhenPush$ והוא יהיה הערך x כאשר נכנס האיבר למבנה הנתונים. כעת ברור גם מה יעשה x- x יסכום את ערכי d שנצטרך לחסר. כלומר, בכל שלב נתון אם נקבל d_1נבצע $x+=d_1$. כעת, כאשר נרצה להכניס איבר למבנה הוא יקבל בשדה שלו את הערך x כאשר נכנס למבנה. כמו כן, גם במקסימומים אנחנו נעדכן את השדה בשביל לדעת כמה עלינו לחסר.

כעת, כאשר נוציא איבר נרצה להחזיר את ערכו + ערך x הנוכחי פחות ערך האיקס שהיה לו בשדה כאשר נכנס וזה ערכו כעת. זהו, המבנה יתמוך בכל הפעולות הנדרשות.

ב. הפרכה - נב"ש שניתן לבצע זאת. ניצור מחסנית כמתואר בסעיף א'. יעלה לנו $O(n)$ על n פעולות $push$. כעת נבצע n פעמים ביצוע מחיקת מקסימום ב$O(1)$ כבנתון. סה"כ קיבלנו מיון מבוסס השוואות ב$O(n)$ בסתירה לחסם תחתון על מיון מבוסס השוואות.

שאלה 4:

אין קשר בין הסעיפים.

א. הוכח הפרך - קיימת ערימת מינימום בינארית $|Q| = n$ כך שמס' הפעולות שיתבצעו בהוצאת המינימום מהערימה בעת מחיקת מינימום יהיה קבוע.

ב. כתוב אלגוריתם שמקבל כקלט מערך A בגודל n עם מספרים טבעיים ומחזיר "קיימת שלשה" אם קיימים שלושה אינדקסים $0 \leq i,j,k \leq n-1$ שונים זה מזה כך שמתקיים $A[k] = 5A[i]$ וגם $A[j] = 2A[i]$. האלגוריתם צריך לרוץ בזמן $O(nlogn)$ או $O(n)$ בתוחלת.

פתרון:

א. הפרכה: נב"ש שקיימת ערימה כזו Q. נבנה אותה בזמן $O(n)$ כבניית ערימה שראינו בהרצאה בזמן לינארי. כעת נניח שמס' הפעולות הקבוע להוצאת איבר המינימום הינה c. נבצע n פעמים הוצאה של איבר המינימום מה שיעלה cn, סה"כ קיבלנו מיון מבוסס השוואות בזמן $cn + n = O(n)$. סה"כ סתירה למיון מבוסס השוואות ב$\Omega(nlogn)$.

ב. פתרון עם תוחלת: נבנה טבלת האש. נרצה לבדוק האם קיימים שלושה איברים ששווים ל$A[i], 2A[i], 5A[i]$. מה שנעשה יהיה לעבור לינארית על איברי המערך. נקח איבר, נבדוק אם קיימים גם $2A[i], 5A[i]$ סיימנו ונחזיר אינדקסים שלהם (בטבלה יוחזקו האיברים עם פוינטר), כך נעבור לינארית על כל איברי המערך. נזכיר שחיפוש בטבלה יעלה $O(1)$ וכן בנייתה ולכן סה"כ $O(n)$

פתרון עם $nlogn$: אמרו $nlogn$ לכן כמובן שנמיין. כי למה לא? כעת נסתכל על $A[1]$ עם פוינטר, על האיבר האמצעי עם פוינטר ועל האיבר האחרון עם פוינטר. נסתכל על האיבר הראשון בתחילה, אם מתקיים שהאמצעי והסוף הם בדיוק $2A[i], 5A[i]$ בהתאמה אזי סיימנו ומצאנו. אחרת נעזר בעובדה שהמערך ממויין, נחפש בחיפוש בינארי את $2A[i], 5A[i]$ מה שיעלה $logn$. אם מצאנו - מבורך וסיימנו. אחרת, נתקדם לאיבר הבא. סה"כ n פעמים כפול $2logn$ פלוס $nlogn$ בהתחלה נקבל $3nlogn = O(nlogn)$

שאלה באינדוקציה:

יהי עץ פיבונאצ'י. עבור $n = 0,1$ העץ הוא צומת בודד. לכל $n \geq 2$ מתקיים שתת העץ השמאלי של השורש הוא T_{n-2} ותת העץ הימני של השורש הוא T_{n-1}.

נסמן ב$T(n)$ את מס' הצמתים בעץ פיבונאצ'י בגודל n.

א. צייר עץ פיבונאצ'י עבור $n=4$.

ב. הראה באינדוקציה על מבנה העץ כי $T(n)=F_{n+2}-1$ כאשר F_i הוא מס פיבונאצ'י הi.

ג. הוכח כי $T(n)\le 2F_{n+2}$

ד. הראה באינדוקציה כי $T(n)\ge F_{n+1}$

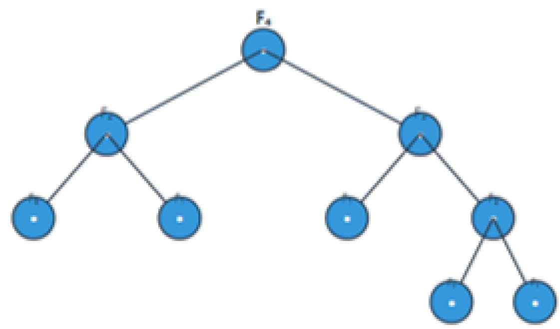
ה. נסמן $L(n)$ מס' העלים בעץ פיבונאצ'י עם n קודקודים. הוכח $L(n)=F_{n+1}$.

ו. הוכח/הפרך: מתקיים לכל $n\in\mathbb{N}$ כי $T(n)=L(n)+T(n-1)$

ז. נסמן $H(n)$ גובה עץ פיבונאצ'י עם n קודקודים. הוכח כי לכל $n\ge 2$ מתקיים $H(n)=n-1$

בונוס (7 נק') : הוכח באינדוקציה כי $\frac{T(n)}{L(n)}\to\phi$.
הדרכה: נתון כי $\phi=\frac{1+\sqrt{5}}{2}$. כמו כן ראינו בתרגול כי $F_n\approx\frac{\phi^n}{\sqrt{5}}$. העזר בסעיפים קודמים.

פתרון:
א.



ב. נוכיח את הטענה באינדוקציה על מס' הצמתים.
בסיס: $n=0$. יודעים כי $F_2=F_1+F_0=2$ וכן אם $n=0$ יש קודקוד בודד ואכן $1=T(n)=2-1$
$n=1$ נקבל $F_3=F_1+F_2=1+1=2$ וכן יש קודקוד בודד ואכן $1=T(n)=2-1=1$
צעד: נניח נכונות לכל עץ עם n קודקודים. נסתכל על השורש. מההגדרה יש לו שני בנים. F_{n-1} וF_{n-2}. כל אחד מהם מקיים את הנחת האינדוקציה ולכן

$$T(n-2)=F_{n-2+2}-1, T(n-1)=F_{n-1+2}-1$$

כמו כן מס' העלים הכולל בעץ הוא מימין+שמאל+1 השורש ולכן

$$T(n)=1+F_n-1+F_{n+1}-=F_n+F_{n+1}-1=F_{n+2}-1$$

ג. לפי סעיף קודם מתקיים $T(n)=F_{n+2}-1$ ולכן שקול להוכיח $F_{n+2}-1\le 2F_{n+2}$, כלומר $-1\le F_{n+2}$ שזו כמובן טענה נכונה כי מס' פיבונאצ'י מקיים $F_n\ge 0$.

ד. כעת עלינו להוכיח $T(n)\ge F_{n+1}$. ובכן לפי ב' שקול להוכיח $F_{n+2}-1\ge F_{n+1}$. שקול לחלוטין מהגדרת פיבונאצ'י להוכיח $F_{n+1}+F_{n+2}-1\ge F_{n+1}$ כלומר $F_{n+2}\ge 1$ שזה כמובן נכון לכל $n\in\mathbb{N}$.

ה. נוכיח את הטענה באינדוקציה.
בסיס: $n=0$ אזי בעץ יש קודקוד יחיד שהוא עלה, ואכן $L(0)=F_1=1$. $n=1$ בדומה בעץ קודקוד יחיד שהוא עלה ואכן $L(1)=F_2=1$.

צעד: נניח שהטענה נכונה לכל עץ פיבונאצ'י עם מקדם $n' < n$. כעת,
סכום העלים בעת הגדול = עלים בבן ימני ועלים בבן שמאלי.
הבן הימני F_{n-1}מקיים הנחת האינדוקציה ויש לו $L(n-1) = F_n$ עלים. הבן השמאלי F_{n-2} גם כן מקיים הנחת האינדוקציה ויש לו $L(n-2) = F_{n-1}$ עלים.
סה"כ בעץ הגדול יש $F_n + F_{n-1} = F_{n+1}$ עלים, כנדרש.

ו. צ"ל כעת לכל $n \in \mathbb{N}$ כי $T(n) = L(n) + T(n-1)$.
לפי סעיף קודם מתקיים $L(n) = F_{n+1}$. לפי סעיף ב', $T(n) = F_{n+2} - 1$ ולכן $T(n-1) = F_{n+1} - 1$.
מכאן ש

$$T(n) = F_{n+1} - 1 + F_{n+1} = 2F_{n+1} - 1$$

ראינו כי $T(n) = F_{n+2} - 1$ ולכן הטענה נכונה אמ"מ

$$F_{n+2} - 1 = 2F_{n+1} - 1$$

כלומר אמ"מ

$$F_{n+2} = 2F_{n+1}$$

הטענה כמובן לא נכונה. עבור $n = 3$ נקבל ש$F_5 = 5$ וכן $F_4 = 3$ וכמובן $5 \neq 2 * 3$.

ז. נוכיח את הטענה באינדוקציה.
עבור $n = 2$ נקבל כי יש להוכיח ש$H(n) = 2 - 1 = 1$. עץ פיבונאצ'י מסדר 2 הוא עץ עם קודקוד, בן ימני ובן שמאלי - גובה עץ כזה הוא אכן אחד כנדרש.
צעד: נניח שהטענה נכונה לכל עץ פיבונאצ'י מסדר $n' < n$. יהי עץ פיבונאצ'י מסדר n. אזי יש לו שני בנים ימני ושמאלי. גובה הבן הימני $H(n-1) = n - 1 - 1$. גובה העץ השמאלי הוא $H(n-2) = n - 2 - 1$.
כמובן גובה העץ המקורי הוא $1 + max\{H(n-1), H(n-2)\} = 1 + max\{n-2, n-3\} = 1 + n - 2 = n - 1$ כנדרש.

ח. נעזר בהדרכה. ראשית נחשב חסם אסימפטוטי עבור $T(n)$.
לפי ב' וה' מתקיים

$$\frac{T(n)}{L(n)} = \frac{F_{n+2} - 1}{F_{n+1}} = \frac{F_{n+2}}{F_{n+1}} - \frac{1}{F_{n+1}} \to \frac{\frac{\phi^{n+2}}{\sqrt{5}}}{\frac{\phi^{n+1}}{\sqrt{5}}} - 0 \to \phi$$

2009 מועד ב' שאלה 1 שמואל
נניח שלכל צומת v בעץ בינארי רשום משקל $W(v)$. נגדיר את המשקל עבור תת עץ כסכום משקלי קודקודיו. כלומר $W(T) = \sum_{v \in T} W(v)$. נסמן ב$L_v$ את תת העץ ששורשו הבן השמאלי של v ובR_v את הימני. עץ בינארי נקרא מאוזן יחסית אם לכל קודקוד בעץ מתקיים התנאי הבא: $W(R_v) < 2W(L_v)$ וגם $W(L_v) < 2W(R_v)$
א. כתוב אלגוריתם יעיל שמקבל עץ T ומחזיר את משקלו $W(T)$
ב. כתוב אלגוריתם יעיל המקבל עץ T ובודק אם הוא מאוזן יחסית
ג. בהינתן שכל משקלות הצמתים חיוביים ממש, מה ניתן להגיד על מבנה של עץ שהוא מאוזן יחסית?

פתרון:
א. נעבוד רקורסיבית. נתבונן בפסודו הקוד הבא ולאחריו אסביר

```
Algo(T)
if T==null return 0
sum.left=Algo(T.left)
sum.right=Algo(T.right)
return T.value+sum.right+sum.left
```

כעת נסביר ונוסיף תנאים: נחשב רקורסיבית, בכל עת יוגדר משקל העץ כערך העץ +ערך בן ימני +ערך בן שמאלי. אם בן ימני ושמאלי לא קיימים כלומר עלה אז ערך העץ הוא בדיוק ערך האיבר. אם קיים רק ימין (לא

ציינתי זאת בפסודו מוסיף כעת) אזי נחשב ערך איבר פלוס ימין, אם קיים רק שמאל בדומה. סה"כ סיבוכיות של $T(n) = 2T(\frac{n}{2}) + O(1) = O(n)$.

ב. שוב נעבוד ברקורסיה. נשתמש בסכומים מקודם לכן ועבור כל צומת בעץ נבדוק אם מתקיים התנאי הדרוש. נתחיל מהשורש - אם לא מתקיים נחזיר שקר. אם כן מתקיים נתקדם רקורסיבית עם הבנים הימניים והשמאליים. כעת לא ברור כל כך מכוונת השאלה האם ייתכן מצב בו בן יחיד ואין בן שמאלי למשל אך נניח שזה כן אפשרי (אם לא אפשרי פשוט נעצור אם הגענו לעלה ונחזיר $true$ ל"וגם" ואז אם יוחזר $true$ מהכל אזי שהעץ מאוזן. אם נאפשר מצב כזה אז ברגע שנגיע למצב בו יש צד ימין בלבד אנחנו נמשיך רקורסיבית ישר עם $true$ כי אין למי להשוות אליו. סה"כ מעבר לינארי על איברי העץ $O(n)$

ג. נניח והעץ מאוזן יחסית. מה שזה אומר זה שהעץ והמשקל בעץ יתאזנו יחסית כלומר האיברים בעץ יהיו יחסית בסדר מאוזן שיתחלק בצורה סבירה, לא ניתן להגיד שהעץ יהיה מאוזן כלומר גובהו לוגריתמי כי אין נתון על מס' קודקודים אלא על סכום ערכים. מה כן אפשר לומר? לכל עץ יש בני בנים - כלומר מדובר בעץ מלא.

2009 מועד ב' שאלה 3 שמואל

נתונות שתי ערימות A וB עם m וn איברים בהתאמה. נניח שקיים k כך ש$m = 2^k - 1$ ושמתקיים $\frac{m}{2} < n \le m$. נרצה לבנות ערימה C שתכיל את כל האיברים של A וB (ניתן להניח שאין איברים משותפים בערימות ושכולם שונים). הערימות נתונות בצורה של עצים בינאריים.

א. מה גובה העץ A ומה גובה העץ B?

ב. מה יהיה עומק C?

ג. תאר אלגוריתם יעיל לבניית הערימה C. מה הסיבוכיות?

פתרון:

א. מתקיים כי $m = 2^k - 1$, כלומר הערימה היא עץ שלם! ידוע כי בעץ שלם מתקיים $m = 2^{h+1} - 1$ ולכן $h + 1 = k$ כלומר $h = k - 1$.

עבור B נראה כי

$$2^{k-1} - 0.5 < n \le 2^k - 1$$

גובה A חסום גם כן ב$k - 1$ בוודאות, ומן הצד השני, הוא גדול מ$k - 2$ ולכן הוא $k - 1$

ב. A היא ממש עץ שלם, וB הוא עץ כמעט שלם ולכן

$$h = 1 + max\{k - 1, k - 1\} = 1 + k - 1 = k$$

ג. בידינו ערימה עם $n + m$ איברים. ידוע כי לבנות ערימה ניתן באופן לינארי עם $O(n + m)$, כמו כן $n \le m$ ולכן סיבוכיות הערימה תהיה $O(2m) = O(m)$.

נוכל לייעל. אם נבצע בניית ערימת מינימום, נשווה בין השורש של A לשל B. את הקטן יותר נוציא מהערימה ונכניס אל C. נחבר את A וB כבניו של השורש בC. כמה פעולות $heapify$ נצטרך לעשות לכל היותר? כגובה הערימה.

2009 מועד א' שאלה 1

נתון דו תור שאורכו בלתי ידוע. אין לנו מצביע לראש או לזנב אלא רק לאיבר מסויים a שלא ידוע לנו מיקומו היחסי ברשימה. מחפשים איבר x שידוע שהוא ברשימה אך לא ידוע היכן הוא ואסור לנו להשתמש במצביע נוסף פרט למצביע הנוכחי שמאותחל בהצבעה על a

א. נבצע חיפוש בצורה הבאה: עבור $k = 1, 2, ...$ עוברים ברשימה החל מהאיבר הa k צעדים שמאלה וחזרה, אח"כ k צעדים ימינה וחזרה עד שנמצא את x. אם נסמן את המרחק (מס' צעדים) בין a לx בn, מה סיבוכיות החיפוש המוצע?

פתרון: במקרה הגרוע ביותר x נמצא באחד מן הקצוות, ואני למשל מאותחל לאמצע הרשימה. במצב כזה נבצע 2 צעדים (רדיוס 1) ואז עוד 4 צעדים (רדיוס 2) וכן הלאה.. עד שנגיע לעשות $2n$ צעדים (רדיוס n)

כלומר נקבל

$$2 + 4 + 8 + ... + 2n = 2\sum_{i=1}^{n} i = 2\frac{n(n+1)}{2} = n(n+1) = O(n^2)$$

ב. כתוב אלגוריתם טוב יותר שמבצע אותו חיפוש בזמן $O(n)$, הנח כי מותר להשתמש במצביעים נוספים.

פתרון: אמנם סעיף ג' פתר לנו גם את ב' אך ניגש אליו גם כן. נשמור מצביע לנקודת ההתחלה, ונתקדם עם מצביע נוסף ימינה לכל היותר n צעדים, או שנתקדם שמאלה עם מצביע אחר n צעדים, לבסוף נמצא אותו ב$O(2n) = O(n)$ צעדים (ניתן תכלס גם לרוץ במקביל עם שני פוינטרים בשני הכיוונים).
פתרון:
ג. כתוב אלגוריתם טוב יותר המבצע חיפוש ב$O(n)$ ללא שימוש במצביעים נוספים. רמז ־ השתמש בסדרה שונה עבור k.
פתרון: הרמז מאוד רמז, אבל העקרון ברור. מה שנעשה יהיה קפיצות אקספוננציאליות, הפעם נגדיר $k = 1, 2, 4, 8, ..., logn$. נקבל כי

$$T(n) = 2\sum_{i=1}^{logn} 2^i = 2\frac{2(2^{log_2 n} - 1)}{2-1} = 4(n-1) = 4n - 4 < 4n = O(n)$$

כעת מדוע זה עוזר לנו להבין את הפתרון? כמו קודם נלך ימינה ושמאלה ברדיוס $logi$, לבסוף נמצא רדיוס שלו ואז עבור איטרציה לאחריו יתקיים $2^{i-1} \leq n \leq 2^i$ כלומר עד $i = logn$ לחיפושו.
הסבר: מבצעים קפיצות אקספוננצאליות עד לגודל קפיצה שהוא לפחות x, ואז יתקיים $2^{i-1} \leq n \leq 2^i$ רדיוס של $logn$ בו נצטרך לחפש בתוכו.

2009 מועד א' שאלה 4
נתונים n איברים שמאוחסנים במערך. בנה עץ חיפוש בינרי רגיל מהם שמכיל אותם (ללא מנגנון AVL)
א. בנה את העץ ע"י הכנסת האיברים אחד אחד לתוך העץ לפי סדר הופעתם במערך. מה סיבוכיות פתרון זה?
ב. הצע אלגוריתם שזמן ריצתו במקרה הגרוע נמוך מהזמן בסעיף א'. מה זמן ריצתו?
ג. האם ניתן לבצע את המשימה ב$O(nloglogn)$?
פתרון:
א. במקרה הגרוע ־ אנחנו נכניס עץ בצורת שרשרת מה שיוביל ל$O(n^2)$ זמן.
ב. נמיין. נבחר את האיבר האמצעי ביותר לשורש העץ. ונבנה רקורסיבית משני הצדדים ־ מה שיעלה לנו $O(nlogn)$ סה"כ.
ג. נב"ש שניתן לבצע את המשימה ב$O(nloglogn)$ זמן. נבצע סריקת $inorder$ שתדפיס את איברי העץ בסדר ממויין, קיבלנו מיון מבוסס השוואות בפחות מ$\Omega(nlogn)$ בסתירה.

2009 מועד א' שאלה 5
עבור עץ בינארי מלא (לכל קודקוד פנימי שני בנים) נגדיר $D(T)$ כסכום עומקי העלים של העץ. כלומר אם $L(T)$ היא קבוצת העלים של T ו$d(v)$ עומק העלה v בעץ אזי $D(T) = \sum_{v\in L(v)} d(v)$.
א. יהי עץ בינארי מלא עם m עלים. נסמן בT_L וT_R את תתי העצים השמאלי והימני של שורשו של T. רשום נוסחה רקורסיבית המבטאת את $D(T)$ כפונקציה של $D(T_L), D(T_R)$.
פתרון:
יודעים כי בעץ בינארי מלא מס' קודקודים פנימיים יהיה $m - 1$. לא ברור אם זה קשור אך נכתוב ונתקדם.
אם יודעים כי בהינתן שורש, אני יודע את סכום עומקי העלים משמאל ויודע את סכום עומקי העלים מימין, אזי סכום עומקי העלים של T יהיה סכומם ועוד סכום העלים בעץ כיוון שכל עלה בעץ יוסיף אחד לעומקו. כלומר $D(T) = D(T_R) + D(T_L) + m$ וכן $D(T) = 0$ אמ"מ T הוא עלה בעצמו.
ב. הוכח באינדוקציה על מבנה העץ כי עבור עץ בינארי מלא T עם m עלים מתקיים $D(T) \geq mlogm$. ניתן להניח כי העץ T הוא מאוזן, וכן הוא מינימלי מכל העצים להם m עלים , בו מס' העלים בשני צדדיו יהיה $\frac{m}{2}$.
פתרון:
בסיס: $m = 1$ יש עלה בודד וזה עץ בודד כלומר עם איבר אחד, עומקו מהנוסחה קודם הוא בדיוק 0 ואכן $1log_2 1 = 1 * 0 = 0$.
צעד: נניח נכונות לכל עץ בינארי מלא T עם $m' < m$ עלים. נסתכל על עץ בינארי T עם m עלים. יש לו בן ימני ושמאלי. שניהם מקיימים את הנחת האינדוקציה כלומר

$$D(T_R), A = D(T_L) \geq \frac{m}{2}log(\frac{m}{2})$$

ולכן סה"כ

$$D(T) = D(T_R) + D(T_L) + m \geq 2 * \frac{m}{2}log(\frac{m}{2}) + m = mlogm - mlog2 + m = mlogm - m + m = mlogm$$

כנדרש.

2006 שמואל מועד ב' שאלה 1

נתונות שתי סדרות ממוינות אחת באורך m והשנייה באורך n. ראינו שניתן למיין אותן בזמן $O(m+n)$

א. הצע אלגוריתם עדיף לאלגוריתם הנתון שירוץ בזמן $O(m+n)$ כאשר $m \leq 2$.

ב. הראה שחסם תחתון לבעיית המיזוג של סדרות ממוינות הוא $\Omega(\binom{m+n}{m})$

פתרון:

א. נפצל למקרים. אם $m=0$ אזי הסדרה שלנו כבר ממוינת ואכן $\Theta(m+n)=\Theta(n)$

אם $m=1$ נרצה להוסיף איבר אחד בדיוק. נסרוק את המערך שאורכו n לכל רוחבו ונבדוק בכל שלב האם x שאני רוצה להכניס מקיים שהוא נמצא בין האיבר שאני בו לאיבר הבא, אם כן נכניסו כלומר ניצור מערך בגודל $n+1$ נעתיק לשם את איברי הרשימה משמאלו, נכניס אותו ואז את איברי הרשימה מימינו. זה אכן מעבר לינארי שיעלה $O(n+1)=O(n)$

אם $m=2$ כמו קודם רק עם שני איברים.

ב. ננסה להבין מה זה הצ'וס המדובר.

$$\binom{m+n}{m}=\frac{(m+n)!}{m!n!}$$

ובכן זה לא מקדם אותנו ולכן נלך לכיוון הקומבינטורי. הנוסחה הזו אומרת, קח סדרה באורך $m+n$, אני רוצה שתחפש m פעמים היכן לשים תו מסויים בתוך הסדרה - וזה אכן חסם תחתון כי חייבים לעבור לינארית על כל האיברים.

2006 מועד א' שאלה 1

נתונות d סדרות ממוינות של מספרים. הסדרות בעלות אורכים שונים וסכום האורכים הוא n. הראה כיצד למזג את d הסדרות לסדרה אחת ממוינת בזמן $O(nlogd)$

פתרון:

נכניס את האיבר הקטן ביותר בכל סדרה אל תוך ערימה שגודלה d. בנייתה תעלה $O(d)$. כעת, בכל שלב נשלוף את המינימום מהערימה, ונכניס אל הערימה את האיבר הבא מהרשימה שכעת הוצאנו את איברה. כלומר, אם הוצאתי איבר x_1 שקושר בפוינטר לאינדקס 1 ברשימה 1, נקדם את הפוינטר לאיבר 2 וכעת נכניס אותו. כך נבצע n פעמים והוצאה מערימה בגודל d וכן הכנסה יעלה יחד $2logd$ ולכן סה"כ $2logd*n+d=O(nlogd)$.

2006 מועד א' שאלה 2

הגדרנו ערימה כעץ בינארי מלא, ולכן מימשנו אותה באמצעות מערך. ניתן לשמור ערימה גם בצורה של עץ רגיל. נניח שנתונות לנו שתי ערימות אחת עם n איברים והשנייה עם m איברים, נניח שהם מיוצגים באמצעות עצים. נרצה לבנות ערימה אחת שתכיל את כל האיברים של שתי הערימות. הראה כיצד לבצע זאת ב$O(log(m+n))$.

פתרון:

נניח כי זו ערימת מינימום. נוסיף ערך חדש שיהיה אנסוף, הבן השמאלי של קודקוד זה יהיה תת ערימה אחת בגודל m והבן הימני בגודל n. נחזיק מצביע לאיבר אנסוף. כעת נבצע פעולת $heapify$ רגילה ואז נמחק ערך זה. לאחר פעולת $heapify$ האיבר המינימלי מבין המינימליים בערימות יעלה אל השורש, בנו השמאלי של השורש יהיה אחד מבניו הקודמים וכן גם מימין יבחר המינימלי מבין השורשרים לעלות לשורש החדש, לכן בנו הימני גם יהיה גדול ממנו. סה"כ פעולת $heapify$ תעלה $max\{logm,logn\}=O(log(m+n))$.

2018 קיץ מועד ב' אלגו 1 שאלה 5

סדרה של הטלות מטבע תקרא תקינה אם אין בה שתי הטלות רצופות של עץ. כתוב אלגוריתם תכנון דינמי אשר בהינתן $n\in\mathbb{N}$ מחשב מס' סדרות תקינות של n הטלות מטבע.

פתרון:

נגדיר $f(i)$ כמס' הסדרות התקינות של הטלת מטבע עבור i מטבעות. כמובן שהפתרון יהיה $f(n)$. נשים לב לתנאי הבסיס הבאים:

אם $i=1$ אזי אורך הסדרה הארוכה ביותר תמיד יהיה באופן ריק כ1.

אחרת, לכל $i>1$ נרצה לבחור מס' אפשרויות ולכן כאן זה יהיה שאלת תכנון דינמי של חיבור ולא מקסום. נשים לב שבהינתן שאני בתא מסויים, אני רוצה להוסיף פלי או להוסיף עץ אם לא היה קודם ולכן נשנה מעט את הנוסחה. נגדיר $f(i,j)$ כאשר i מוגדר כמו קודם ו$j\in\{0,1\}$ כאשר 0=עץ ו1=פלי, כלומר הנוסחה תחשב את מס' הסדרות התקינות עבור $1,...i$ מטבעות כאשר **בבחירה הנוכחית בחרתי** j. כעת קל יותר לחשב את הנוסחה כי בהינתן ש$j=0$

אזי בחרתי כעת עץ ולכן נרצה לסכום רק מס' אפשרויות ממקודם לכן שניתן היה להגיע אליהן עם עץ. אחרת, בחרתי $j=1$ כלומר לא אכפת לי ממקודם לכן מהיכן הגעתי, אם זה אפס או אחד.

$$f(i,j) := \left\{ \begin{array}{cc} 1 & i=1 \\ f(i-1,1) & i>1 \wedge j=0 \\ f(i-1,1)+f(i-1,0) & i>1 \wedge j=1 \end{array} \right\}$$

כעת לתכנון הדינמי ניצור מטריצה $n \times 2$. נמלאה לפי נוסחת הנסיגה בעמודות. מילוי תא יעלה $O(1)$, ויש $2n$ תאים ולכן סיבוכיות הזמן תהיה $O(n)$, את המקום נוכל להוריד ל$O(1)$ אם נשמור בכל שלב שני עמודות קודמות. סה"כ הפתרון יהיה $f(n,1)+f(n,0)$.

2024 תל אביב שאלה 3

עבור מערך $A[1,...n]$ המכיל מס' ממשיים נסמן $m_A = min_{1\leq i\leq n} A[i], M_A = max_{1\leq i\leq n} A[i]$.

א. הוכיחו כי קיימים $1 \leq i \neq j \leq n$ כך ש $0 \leq A[i]-A[j] \leq \frac{M_A-m_A}{n-1}$

רמז: הוכיחו כי במערך B אשר מתקבל ע"י מיון מערך A קיים $1 \leq i \leq n-1$ עבורו $0 \leq B[i+1]-B[i] \leq \frac{M_A-m_A}{n-1}$

פתרון:

נעזר ברמז ונמיין את מערך A. בהכרח נקבל $B[n]=M_A$ וכן $B[1]=m_A$. כלומר, צ"ל כי קיימים $0 \leq B[i+1]-B[i] \leq \frac{B[n]-B[1]}{n-1}$

נשים לב שאם הוכחנו את הטענה כמובן שהוכחנו אותה עבור A כי הם אכן קיימים בA.

נב"ש שלא קיים אינדקס כזה i. כלומר, לכל i במערך B מתקיים $B[i+1]-B[i]<0$ ואז נקבל $B[i]>B[i+1]$ בסתירה.

או ש $\frac{B[n]-B[1]}{n-1} < B[i+1]-B[i]$. בפרט זה יתקיים לכל i ולכן

$$\sum_{i=1}^{n-1} B[i+1]-B[i] > (n-1)\frac{B[n]-B[1]}{n-1} = B[n]-B[1]$$

נראה כי הסכום המדובר משמאל הינו סכום טלסקופי.

$$B[2]-B[1]+B[3]-B[2]+....+B[n]-B[n-1] = B[n]-B[1]$$

כלומר קיבלנו סה"כ

$$B[n]-B[1] > B[n]-B[1]$$

והרי שזו סתירה כיוון שזה ממש שווה.

ב. כתוב אלגוריתם שמקבל כקלט מערך $A[1,...n]$ ומוצא זוג אינדקסים $1 \leq i \neq j \leq n$ שמקיימים $0 \leq A[i]-A[j] \leq \frac{M_A-m_A}{n-1}$, זמן ריצת האלגוריתם $O(n)$

פתרון:

נסמן את הערך $t = \frac{M_A-m_A}{n-1}$, שיהיה יותר קל לעבוד, כעת צריך למצוא $0 \leq A[i]-A[j] \leq t$ בזמן $O(n)$ והרי שקיימים כאלו מסעיף א'. את הערך t נוכל לחשב בקלות ע"י הפעלת סלקט (או פשוט סריקה של מקסימום ומינימום במערך, יעלה $O(n)$)

נשתמש במושג שנקרא בקט - "תא " אליו נשייך איברים

אנחנו יודעים את טווח המספרים של כל אחד מהאיברים $m_A \leq A[i] \leq M_A$. נפרוס את הקטע ל$n-1$ קטעים, כל אחד שווה ברוחבו, למשל אם יש לנו טווח מספרים $[1,99]$ ויש 4 מספרים, נחלק את המקטע ל4 בקטים, $[1,33],[34,66],[67,99]$ שווים באורכם. כעת לכל איבר נשלח אותו לבקט מסויים לפי הנוסחה $\lfloor \frac{x-m_A}{t} \rfloor$ כאשר t הוא הערך בו דנו קודם וm_A הוא המינימום של המערך. כך למשל האיבר 45 בתוך המערך $[1,45,58,99]$ בו $t=\frac{99-1}{4-1}=32.66$ יכנס לבקט $1=\lfloor \frac{45-1}{32.66} \rfloor = \lfloor \frac{x-m_A}{t} \rfloor$. כעת לאחר שביצענו פעולות אריתמטיות שעלו לכל היותר $O(n)$, יש לנו $n-1$ מקטעים, לפי סעיף א' ושובך היונים קיים איבר שנמצא בשניהם, ולכן נסרוק את הבקטים פעם אחרונה למצוא את הבקט עם שני האיברים (יתכנו יותר, נעצור כאשר נמצא את האחד הראשון).

שאלה 4 2024 תל אביב:

תארו מבנה נתונים שמתחזק קבוצת איברים מתחום בעל סדר מלא ותומך בפעולות הבאות כאשר n מתאר גודל הקבוצה:

א. הכנסת איבר x למבנה $O(logn)$
ב. מחיקת איבר x בהינתן מצביע לייצוגו במבנה $O(logn)$
ג. מחזירה האם האיבר x במבנה $O(logn)$
ד. $AlternaitingDelations(k)$פעולה שתמחק את כל האיברים בעלי דרגה i כך ש$\frac{i}{k}$ בערך עליון אי זוגי. כלומר מוחקים את k האיברים הקטנים ביותר, את הk הבאים משאירים, הk הבאים מוחקים וכן הלאה. סיבוכיות:
א. הצע מימוש שירוץ בזמן $O(n)$ ללא קשר לערכו של k
ב. $k=\sqrt{n}$ הצע מימוש שירוץ בזמן $O(\sqrt{n}logn)$

פתרון:
נשתמש כמובן בעץ AVL ונקבל הכנסה מחיקה וחיפוש ב$O(logn)$. כעת הפעולה היחידה הבעייתית היא ד'. נראה כי בהינתן ערך k כלשהו נהיה תמיד חייבים למחוק חצי מאיברי המבנה. מה שנעשה יהיה סריקה רוחבית של העץ, באמצעות $inorder$. אנחנו נסרוק את העץ ונקבל את הסדר הממויין שלו מה שיעלה לנו $O(n)$. כעת, בהתאם לערך k נקח את המערך הממויין שקיבלנו ממנו ונשאר עם $\frac{n}{2}$ איברים במערך ממויין חדש. כעת, נבנה רקורסיבית ממנו עץ AVL חדש בזמן $O(n)$ כי הוא ממויין אז ניתן לבנות רקורסיבית באמצעות בחירת האמצעי ואז הלאה.
סעיף ב -
כעת נרצה לשפר סיבוכיות, נניח ונתון כי $k=\sqrt{n}$, גם כאן נרצה למחוק את חצי מהאיברים. כעת, נרצה שישארו $\sqrt{n}$ איברים קטנים ביותר, ואח"כ למחוק $\sqrt{n}$ איברים, ואח"כ $\sqrt{n}$ הבאים בגודלם להשאיר.
בהינתן k קבוצות של $\sqrt{n}$ מתקיים $k\sqrt{n}=n$, כלומר $k=\sqrt{n}$, כלומר יש לנו $\sqrt{n}$ חלוקות כאלו. כמה חלוקות נרצה למחוק? בערך $\frac{\sqrt{n}}{2}$ שבכל אחת $\sqrt{n}$ איברים.
ננסה להבין מדרישת הסיבוכיות - יש לנו שורש אן פעמים כפול לוג אן, כלומר אנחנו צריכים לבצע שורש אן פעמים כפול משהו בעץ.
נוסיף לעץ שדה $rank$ - כמה קטנים ממני. ניתן לתחזק זאת בזמן העץ עם $O(1)$ זמן תוספת כלומר קבוע. נבצע $split$ (נחפש נקודה לפיצול ונפצל את העץ לשני עצים עצמאיים ב$O(logn)$) לאיבר השורש אן-י בגודלו, ואח"כ שוב לאיבר השורש אן-י בגודלו, ובסוף נעשה $join$ איחוד שני עצים מאוזנים לעץ אחד) לכל העצים, סה"כ כפול שורש אן עצים נקבל סיבוכיות $O(\sqrt{n}logn)$.

שאלה 5 תל אביב 2024:
מעוניינים במבנה נתונים שמתחזק קטעים על ישר. כל קטע מהצורה (a,b) עבור $a<b$ ממשיים. האורך של (a,b) הוא מס' חיובי $L=b-a$. ונקודת האמצע שלו היא המספר $\frac{a+b}{2}$. הקטע (a,b) שמאלי יותר מהקטע (c,d) אם $a<c$ וכן ימני יותר אם $b>d$. שימו לב כי יתכן שקטע הוא גם שמאלי וגם ימני יותר מקטע אחר.
א. $insert(a,b)$ מקבל $a<b$ ממשיים ומכניס למבנה קטע חדש (a,b) ומחזיר מצביע אליו. $O(logn)$
ב. $delete(I)$ מקבל מצביע לקטע I במבנה ומוחק אותו $O(logn)$
ג. $Left()$ מחזיר מצביע לקטע השמאלי ביותר במבנה $O(1)$
ד. $right()$ מחזיר מצביע לקטע הימני ביותר במבנה $O(1)$
ה. $Largest()$ מחזיר מצביע לקטע הארוך ביותר במבנה $O(1)$
ו. $Expand(I,c)$ מקבל מצביע לקטע I ומס' ממשי $c>0$ ומרחיב את הקטע כך שאורכו יגדל בc ללא שינוי נקודת האמצע שלו. $O(logn)$

n הוא מס' קטעים במבנה. שימו לב ניתן להניח שלאורך כל השימוש במבנה הנתונים לא נתקל באותו המספר הממשי יותר מפעם אחת.

פתרון: נראה כי המשפט בסוף חשוב מאוד. זה אומר שלא יהיה לנו חפיפה בקטעים. כמו כן נסתכל על ו' ונראה כי אם ירצו מאיתנו שהאורך יגדל בc ללא שינוי נקודת האמצע, נהיה חייבים להוסיף $\frac{c}{2}$ מימין ומשמאלו. כלומר לקבל את הקטע $[a-\frac{c}{2},b+\frac{c}{2}]$.
כעת, נרצה לחשוב במה להשתמש עבורו.
נראה כי הקטע השמאלי ביותר = נקודת הa שלו הכי קטנה. הקטע הימני ביותר = נקודת הb שלו הכי גדולה. הקטע הארוך ביותר $l=b-a$ מקסימלי ביחס אליו.
נשתמש בשלושה עצי AVL. עץ אחד ישמור את ערך הa של הנקודה, עץ שני את ערך הb של הנקודה ועץ שלישי את אורך הנקודה L. הם יקושרו אחד לשני הערכים באמצעות מצביעים. נסמן את העצים כT_a,T_b,T_L.
הכנסה: נכניס את a לT_a את b לT_b ואת L לT_l באמצעות פוינטרים. הכנסה לעץ תעלה $O(logn)$ ולכן סה"כ $3logn=O(logn)$.
הוצאה: יקבל מצביע לקטע עם (a,b). נחשב את l ב$O(1)$, נחפש אותו בעץ T_l ונמחק אותו משם, אח"כ נחפש את a וb ונמחק אותם מT_a וT_b בהתאמה. סה"כ לחיפוש והוצאה $O(logn)$ ולכן $6logn+1=O(logn)$.
שמאל: נרצה לחפש את הקטע שנקודת הa שלו הכי קטנה, אנחנו מאחסנים בT_a את הערכים בעץ חיפוש ולכן הערך הנמוך ביותר יהיה ללכת שמאל עד הסוף כגובה העץ $O(logn)$ וזה לא טוב לנו. רצו $O(1)$, לכן נחליט שנוסיף לעץ T_a שדה של האיבר המינימלי ביותר בתת העץ. זה יעלה לתחזק $O(1)$ בפעולות השונות, ולכן כאשר נרצה את

הקטע השמאלי ביותר, נגש לשורש של T_a ונשלוף משם את הערך המינימלי ביותר בתת העץ השמאלי, נקבל את הערך a, כמו כן שמרנו אליו גם מצביע, ומהמצביע של a מקושר לנו המצביע לקטע של b, ואז נגיע לb בקטע השני ונחזיר (a,b)

ימין: ממש כמו שמאל רק שנחזיק את הערך המקסימלי בתת העץ ואז נרצה את הערך המקסימלי של השורש מימין, נגיע עם המצביע אל b ואז עם המצביע שלו נגיע אל a בקטע השני

הכי ארוך: בדומה, נחזיק שדה מקסימלי לאורך הקטע, נרצה את המקסימום של השורש, נשתגר עם מצביע לאורך הקטע ולו יש מצביעים לa,b ונוכל לדעת אותם (מספיק יהיה גם להשתגר עם מצביע אחד לערך אחד מהם ואז לחשב את השני בהתאם בהתחשב לכך שיודעים אורך קטע)

הרחבה: הרעיון יהיה כזה - נחפש את האיבר a בתוך T_a ואת b בתוך T_b. נראה כי בשביל לשמור על נק' האמצע נצטרך שהקטע יהפוך לקטע $[a-\frac{c}{2},b+\frac{c}{2}]$, כעת אורך הקטע יהפוך להיות $b+c-a=L+c$. כלומר הגדלנו את אורך הקטע בc בדיוק. מה שנעשה יהיה להוציא את אורך הקטע l מT_L, ולהכניסו עם מצביע לa,b מחדש עם הערך $L+c$. בדומה, נוציא את b ונכניסו לעץ T_b עם $b+\frac{c}{2}$ ואת a נוציא ונכניסו חזרה עם $a-\frac{c}{2}$. נתון בשאלה כי לא נתקל באותו מס' ממשי ולכן אין צורך לבדוק בעיות מסוג זה, סה"כ ביצענו 9 הכנסות חיפושים והוצאות שעולות $O(logn)$ ולכן $9logn=O(logn)$.

שאלה 2 תל אביב $2024BA$

תארו מבנה נתונים שתומך בפעולות הבאות כאשר n מתאר את מס' האיברים במבנה

א. $insert(k,v)$ הכנסת איבר חדש למבנה עם מפתח k וערך v, שניהם מספרים כלשהם. ניתן להניח שאיבר עם מפתח k לא קיים כבר סיבוכיות $O(logn)$

ב. $search(k)$ החזרת מצביע לצומת בעל מפתח k או הודעה אם לא קיים $O(logn)$

ג. $delete(x)$ מחיקת איבר x בהינתן מצביע לייצוגו במבנה $O(logn$

ד. $MaxRange(a,b)$ החזרת הערך v המקסימלי מבין האיברים בעלי מפתח k בתחום $[a,b]$.

ממשו את הפעולה האחרונה בסיבוכיות זמן הטובה ביותר שתוכלו.

פתרון:

נשתמש כמובן בעץ AVL עם מפתח k וערכי v כאשר נקבל הכנסה הוצאה ומחיקה ב$O(logn)$ כנדרש. כעת נרצה להוסיף את הפעולה האחרונה. לשם כך נרצה להוסיף שדות לעץ. נראה כי סיבוכיות הזמן הטובה ביותר שנוכל לתת היא $O(logn)$ כי נהיה חייבים לבצע חיפוש של a,b בעץ תחילה. כלומר נבצע חיפוש של מפתחות a,b וכעת אנחנו יודעים את התחום בו אנחנו רוצים לפעול.

לבצע זאת ב$O(logn+k)$ כאשר k מס' מפתחות בתחום זה קל כי נעבור עליהם. נרצה להראות שבהינתן שיש לי כעת שני ערכי מפתחות אני יודע את המקסימום הלוקלי שלהם ב$O(1)$

זה יהיה קצת קשה לכן ניגש לזה אחרת. נרצה להוסיף שדה מקסימום בתת העץ. קל להוסיף שדה זה עם $O(1)$ עבודה ע"י תחזוק בזמן גלגולים. נחזיק שדה מקסימום בתת עץ ימני ומקסימום בתת עץ שמאלי וכמובן המקסימום של תת העץ יהיה מקסימום תת עץ ימני כי ימין> שמאל בעצי חיפוש.

יהי r שורש העץ.

אם $a>key(r)$ נלך ימינה כי גם בהכרח b שם. אם $b<key(r)$ נלך שמאלה כי גם בהכרח a קטן משורש העץ. אחרת, כלומר $a\leq key(r)\leq b$ כלומר המפתחות שלנו נמצאים בין קצוות העץ.

אם נקבל שתת עץ שלם נמצא בתוך הטווח שלנו, נחזיר את המקסימום שלו - הערך ששימרנו. הוא המקסימום הלוקלי.

אחרת, אם לא כל תת העץ בטווח, נרד לחפש רקורסיבית בתתי העצים.

א. תת עץ שמאלי -

1. אם מקסימום תת העץ השמאלי קטן מa דלג על תת העץ השמאלי לגמרי.
2. אם כל תת העץ השמאלי ב$[a,b]$ קח את המקסימום שלו.
3. תמשיך רקורסיבית על תת העץ השמאלי

ב. תת עץ ימני

1. אם המינימום של תת העץ הימני גדול מb, דלג עליו לגמרי.
2. אם כל תת העץ הימני ב$[a,b]$ קח את המקסימום שלו
3. אחרת תמשיך רקורסיבית על תת העץ הימני.

התוצאה תהיה המקסימום המקומי שהוחזר מהרקורסיה. סה"כ רקורסיה $O(logn)$ על גובה העץ

שאלה 3 $2024BA$

א. תארו מבנה נתונים שמאחסן מספרים שלמים ותומך בפעולו הבאות:

א. הכנסת איבר חדש למבנה עם מפתח k טבעי הנח שלא קיים כבר $O(logn)$

ב.חיפוש מפתח k $O(logn)$

ג. מחיקת מפתח k $O(logn)$

ד. מחיקה ממבנה הנתונים חצי מהמפתחות שמתחלקים ב7 . המפתחות שיימחקו מבין המתחלקים ב7 שרירותיים לבחירתכם $O(logn)$.

פתרון:

נשתמש כמובן בעץ AVL ונקבל את א+ב+ג בחינם. כלומר, כמו שלמדנו בהרצאה ב$O(logn)$

כעת נרצה להתמקד בפעולה האחרונה והמעניינת. ראשית נצטרך לענות על השאלה - כיצד אני יודע מי מתחלק ב7? שנית, בהינתן שאני יודע מיהם המתחלקים ב7 - כיצד אני מוחק חצי מהם? הרי אם כל המפתחות מתחלקים ב7 עליי למחוק חצי מהעץ. זה לכל הפחות עולה $O(n)$ ע"י סריקה ובנייה מחדש, או $O(nlogn)$ אם אני ממש מוחק אחד אחד.

בעת הכניסה למבנה הנתונים נבדוק האם המספר מתחלק ב7. במידה וכן, נשלח אותו לעץ המתחלקים ב7. עץ כזה יהיה בגובה של לכל היותר $O(logn)$ עם n איברים במקרה הגרוע. נרצה למחוק חצי מהם, ובכן נעשה $split$ על האיבר האמצעי בעץ, מה שיעלה לנו $O(logn)$ לפצל את העץ לשניים. כעת נשארנו עם מחצית מהמספרים שמתחלקים ב7, ב$O(logn)$.

שאלה 1 2024 תל אביב

להלן תור משכולל השומש אוסף איברים בשיטת $FIFO$:

א. $enqueue(x)$ מכניס x לסוף התור

ב. $dequeue(x)$ מחזיר איבר שבראש התור ומוחק אותו.

ג. $retrieve(i)$ מחזיר איבר הi הראשון בתור, $i=1$ הוא הראשון בתור, $i=n$ הוא האחרון בתור.

הצע מבנה נתונים שתומך בזמן אמורטייזד $O(1)$ לכל אחת מהפעולות.

פתרון:

נראה כי הפעולה היקרה עבורנו היא ג', שכן היא דורשת ממש מידע על כל איבר.

הפתרון יהיה שימוש במערך דינמי. נתחזק מערך דינמי עם מצביע לראשו ומצביע לזנב שלו. בכל פעם כאשר המערך הדינמי יתמלא, אנחנו נכפיל את גודלו פי 2. נעתיק אליו את האיברים כמובן. כאשר נקבל כי המערך הדינמי מגיע למחצית מהגודל הדרוש שלו, כלומר מחקנו יותר מדי (נגדיר מס' קבוע עבורו נדע מה הרף ש"אסור" לרדת ממנו) אזי נקטין את המערך פי 2 כלומר ניצור חדש בגודל חצי מהקודם ונעתיק אליו את האיברים. ראינו כבר הן בתרגול והן בהרצאה שניתן לבצע מערך דינמי ב$O(1)$ זמן, והתעכבנו על זה ארוכות מדוע זה $O(1)$ לשיעורין.

כעת, נתאים את הבעיה לשאלה שלנו.

מה שנרצה לעשות יהיה לתחזק מערך דינמי, וקאונטר שיספור את מס' האיברים כעת במערך שלנו. כאשר נרצה להחזיר את איבר בראש התור ולמחוק אותו, זה האיבר הראשון בתור למעשה כרגע, נמחק אותו נוריד את הקאונטר ונקדם את המצביע. כאשר נרצה להוסיף איבר בסוף התור נכניס במיקום האחרון אליו מצביע המצביע שלנו (אחד אחריו, נקדם את המצביע ונעלה את הקאונטר)

כעת נשים לב שלהגיד עבור הפעולה השלישית "נדפיס איבר $A[i]$ " זה לא נכון. יתכן שאנחנו במצב שבו מחקו לנו הרבה מההתחלה למשל וזה לא אכן כך הדבר. מה שנעשה יהיה להסתכל על האינדקס בו מתחיל התור שלנו, יש לנו מצביע לאיבר זה, ואז אם נרצה למשל את האיבר הi בתור הוא יהיה במיקום $i+indexOfStart$, שכן לא ניתן למחוק מאמצע התור.

פעולה זו כמובן ב$O(1)$, ובאמצעות הניתוח לשיעורין מההרצאה אודות מערך דינמי כעת חסך לנו עבודה בהסבר מתחילת השאלה.

שאלה 2 2017 מועד א' אלגו 1

השאלה: נניח שבקלט לאלגוריתם הופמן קיים תו שתדירותו היא יותר מ$\frac{2}{5}$. אזי חייבת להיות מילת קוד באורך אחד.

הוכחה: יש לנו תו s עם תדירות> $\frac{2}{5}$. כל שאר התווים יחד, סכום התדירויות שלהם יהיה קטן מ$\frac{3}{5}$. נוכיח שs יקבל מילת קוד באורך 1.

נב"ש כי c יקבל מילת קוד באורך $c\geq 2$. באלגוריתם הופמן מיזוג קורה רק בין שני הצמתים עם התדירות הנמוכה ביותר. בכל שלב של האלגוריתם $f(s)>\frac{2}{5}$, וכן סכום התדירויות $\sum -f(s)<\frac{3}{5}$. לכן כל צומת אחר (בודד או ממוזג) תהיה מקסימלית עם תדירות $<\frac{2}{5}$. אם $f(s)>\frac{2}{5}$ וכל צומת אחר קטנה מ$\frac{3}{5}$ אזי באף שלב דוכל חיבור אחר לא יוכלו להיות השניים הקטנים ביותר. לכן s לא יתמזג עד הסוף ולכן עומקו יהיה 1 כלומר תהיה מילת קוד באורך 1. הסבר: בשביל שs יתמזג עם מישהו, צריך שתהיה צומת אחרת שהתדירות שלו תהיה פחותה או שווה לתידורת של s אבל כל צומת אחר מורכב מערכים עם סך תדירויות קטן משלוש חמישיות ולכן אפילו אם נחבר את שני הצמתים הכבדים ביותר שאינם s התוצאה תהיה $\frac{3}{5}$. סה"כ התו לא יתמזג עם עוד איבר נוסף ולכן ישאר בודד כלומר מילה באורך 1.

אלגו 1 2015 תרגיל 6 שאלה 3

חנות "הצילום לכל" מתמודדת עם בעיית תזמונים הבאה: כל יום בבוקר מתקבלות עבודות מלקוחות שונים לביצוען במהלך היום על מכונת הצילום היחידה של החנות. לא ניתן לבצע שתי עבודות בו זמנית. מטרת החנות היא למקסם שביעות רצון לקחותיה ע"י קביעת לוח זמנים טוב ביותר לביצוע העבודות.

באופן פורמלי לחנות נתונות בבוקר יום n עבודות של לקוחות לביצוע עבודתו של הלקוח הi נמשכת t_i זמן. כמו

כן לכל לקוח i יש ערך חיובי w_i שמציין את מידת חשיבותו לחנות. זמן סיום לקוח i הוא c_i. מידת השביעות הרצון של הלקוח תלויה בזמן הסיום של ביצוע העבודה עבורו.
החנות רוצה שהסכום המשוקלל $\sum_{i=1}^{n} w_i c_i$ יהיה מינימלי.
כתוב אלגוריתם חמדן לבעיה והוכח נכונות.

פתרון:
נשים לב כי אם נסתכל על הערך $\frac{w_i}{t_i}$ ונמיין זה בדיוק כמו בבעיית תרמיל הגב בשלמים. שם הוכחנו שאם נסתכל על הערך של העלות שלו שהוא תורם (v_i שם כאן w_i) חלקי כמה הוא דורש ממני (כאן t_i שם w_i) ונמיין, ובכל שלב נקח את הבעיה שהערך הזה שלה הוא הגדול ביותר, נגיע לפתרון אופטימלי.
כעת ניגש להוכחת נכונות:
א. למת הבחירה החמדנית - קיים פתרון אופטימלי לבעיית התזמונים אם נבחר בכל פעם את הערך הגבוה ביותר ביחס הניתן.
הוכחה: נניח שנקבל רשימה של השברים שחילקנו ממוינת. נסתכל על האיבר שיחסו הוא הגדול ביותר, עם זמן t_i וערך חשיבות w_i. כמו כן נסתכל על הפתרון האופטימלי OPT. אם $i \in OPT$ סיימנו. אחרת, נסמן את התחנה שנבחרה במקום להיות בתוך הפתרון האופטימלי כj. נשים לב שכמובן יתכנו הרבה כאלו אך בהכרח נסתכל על זו שנבחרה ראשונה (ואז רקורסיבית, אם היא לא טובה נעיף מישהי בסוף). בהכרח מתקיים $\frac{w_i}{t_i} > \frac{w_j}{t_j}$. לכן נסתכל על הפתרון

$$OPT' = OPT/\{j\} \cup \{i\}$$

ראשית - ברור שהיא חוקית. אופטימלית, נרצה לשים לב כי

$$|OPT'| = |OPT/\{j\} \cup \{i\}| = |OPT| - w_j c_j + w_i c_i \geq |OPT|$$

כעת נשים לב כי $w_i c_i \geq w_j c_j$. מדוע? בחרנו מראש את האופטימלי ביחס לזמן ולערך, ולכן בהכרח זה מתקיים ומשם השוויון מעלה.
כלומר אכן גודל הפתרון קטן שווה מהאופטימלי, ולכן בהכרח אופטימלי בעצמו וחוקי.

תכונת תת המבנה האופטימלי: פתרון שמורכב מבחירה של הערך השבר הגדול ביותר בתוספת פתרון לתת הבעיה של האיברים לאחר מכן, הוא בהכרח פתרון אופטימלי לבעיה.
נוכיח.יהי פתרון אופטימלי A. כעת, נניח בשלילה שהפתרון של האלגוריתם A איננו אופטימלי. כלומר קיים B פתרון כך ש$|A| < |B|$. עקב הבחירה החמדנית בהכרח $i \in B$. היא הראשונה בדרך. בפרט $A/\{i\}$ וכן $B/\{i\}$ הם פתרונות אופטימליים לבעיה ללא הערך הראשון שסומן כi.בפרט, $A/\{i\}$ פתרון אופטימלי. כלומר

$$|A/\{i\}| \leq |B/\{i\}| \Longrightarrow |A| - w_i c_i \leq |B| - w_i c_i \Longrightarrow |A| \leq |B|$$

בסתירה ל$|B| < |A|$. סה"כ אכן נקבל פתרון אופטימלי.
זמן ריצה - למיון $O(nlogn)$ ואז מעבר לינארי $O(n)$ ולכן סה"כ $O(nlogn)$.

2007 מועד ב' שאלה 5 שמואל
הוכח באינדוקציה על מבנה העץ שלעץ בינארי שגובהו h יש לכל היותר $2^{h+1} - 1$ קודקודים.
פתרון:
בסיס: $h = 0$ כלומר קודקוד בודד, אכן $2^{0+1} - 1 = 1$.
צעד: נניח נכונות לכל עץ מגובה $h' < h$. יהי עץ בינארי מגובה h. אזי, נחלק למקרים:
א. יש לעץ שני בנים, T_R, T_L. בהכרח הגובה שלהם יהיה קטן מh (לכל היותר $h - 1$), ולכן מקיימים את הנחת האינדוקציה. כלומר

$$n_r \leq 2^{h_r+1} - 1, n_l \leq 2^{h_L+1} - 1$$

כעת, גובה העץ כולו יהיה

$$h(T) = 1 + max\{h(T_L), h(T_R)\}$$

בה"כ נניח שהמקסימום הוא גובה תת העץ הימני. כעת נראה כי אם נסמנו h_r והוא המקסימום, בהכרח $h_r+1 = h$, כלומר $h_r = h-1$, וכן $h_L \leq h_r = h-1$. כעת נראה כי מס' הקודקודים בעץ כולו יהיה מס' הקודקודים מימין +משמאל ועוד אחד ונקבל

$$n \leq 1 + 2^{h_r+1} - 1 + 2^{h_L+1} - 1 = 2^{h_r+1} + 2^{h_L+1} - 1 \leq 2 * 2^{h_r+1} - 1 = 2^{h_r+2} - 1 = 2^{h+1} - 1$$

אם כן, הטענה הוכחה.

ב. יש לעץ בן שמאלי בלבד (בה"כ. המקרה עבור בן ימני בלבד סימטרי)

אזי גובהו בהכרח קטן מh והוא מקיים הנחת האינדוקציה. במקרה זה, גובהו $h_l = h-1$. ולכן $n_l \leq 2^{h_L+1} - 1$, אם כן מס' הקודקודים בעץ כולו יהיה מס' הקודקודים בעץ השמאלי ועוד אחד ונקבל

$$n \leq 1 + 2^{h_L+1} - 1 = 2^{h_l+1} = 2^h \leq 2^{h+1}$$

2025 מועד ב' שאלה 2

יהיו T_1, T_2 עצים שאינם מיוצגים במערך ערמות מקסימום בינארית בגודל n.

א. הצע אלגוריתם יעיל ביותר לאיחוד העצים לערימת מקסימום אחת שאינה מיוצגת במערך.

פתרון: נבצע סריקה על שני העצים ונכניסם למערך עזר בגודל $2n$. זה יעלה $O(2n)$. נבנה ערימה מהמערך. כעת, נבנה באופן רקורסיבי חזרה עץ מהמערך ע"י צומת i עם בנים $2i+1, 2i+2$. סה"כ $O(n)$.

ב. נניח $n = 2^k$. כאשר k טבעי. הצע אלגוריתם יעיל ביותר לאיחוד העצים.

פתרון:

מהנתון נבין כי הערימה היא עץ שלם לגמרי, לא רק כמעט שלם וכן היא עץ מלא לחלוטין. כעת נרצה לקחת את המקסימום מבין הקודקודים שבראש הערימות, ואת השאריות משתי הערימות נשים כבנים של המקסימום. כעת כיוון שהם עץ שלם לגמרי, נצטרך רקורסיבית לתקן ב$O(logn)$ סה"כ.

כעת נסביר יותר טוב כיצד יבנה העץ הזה: נשווה את שני השורשים, הגדול מהם יהיה המקסימום בעץ ואז שני העצים יהיו בניו. נבנה עץ חדש כך: שורש חדש ־ מקסימום שנבחר, בן שמאלי = תת עץ שמאלי של שורש שנבחר. בן ימני = מיזוג רקורסיבי בין בן ימני של שורש שנבחר לבין העץ שני כולו . בכל שלב ישאר המצב בו האב גדול משני בניו וכן נראה כי בכל שלב גודל הקלט יורד באחד בגובה, כלומר חצי בגודל הקלט ולכן $T(n) = T(\frac{n}{2}) + O(1) = O(logn)$.

2025 שאלה 3 מועד ב'

נאמר כי אריח הוא בצורת האות L אם הוא ריבוע בגודל 2×2 עם תא חסר בגודל 1×1. קלט יהיה לוח בגודל $n \times n$ כאשר $n = 2^k$. $k > 1$. בלוח תא אחד חסר בגודל 1×1.

פלט: מיקום אריחים לכיסוי כל הלוח (למעט תא חסר) באריחים בצורת האות L. הבעיה ניתנת לפתרון בהפרד ומשול.

פתרון:

בדיוק כמו הבעיה שראינו בתרגול ובבדידה. נרצה לצמצם את תת הבעיה שלנו ל4 תתי בעיות. לכן מה שנעשה יהיה להסתכל על החלק בו הם מתפצלים ל4 חלקים, לסמן בשחור 3 משבצות ברבעים שאין בהם את התא השרוף. סה"כ כך נמשיך למלא את המטריצה רקורסיבית ונקבל $T(k) = 4T(\frac{k}{4}) + O(1)$. פתרון הנוסחה הוא $O(k)$.

הנוסחה למעשה היא $T(n^2) = 4T(\frac{n^2}{4}) + O(1)$ ואם נציב $k = n^2$ נקבל את סיבוכיות הנוסחה מקודם לכן $O(k) = O(n^2)$

2025 מועד ב' שאלה 4

קלט: מערך A לא ממויין של מספרים שלמים בגודל n ומס' k

פלט: אם קיים תת מערך של A שסכומו שווה לk החזר אמת אחרת שקר.

הצע אלגוריתם יעיל ככל הניתן לפתרון הבעיה.

פתרון: נבנה מערך סכומים חלקיים $S[i] = \sum_{k=1}^{i} A[k]$ ב$O(n)$ בצורה רקורסיבית עם התא הקודם. כעת, עבור כל ערך i נסתכל על $A[i]$. נראה כי אם קיים ערך k שכזה, הוא באינדקסים $i, ..j$. וסכומו יהיה $S[i] - S[j-1]$. כעת עבור כל i, נרצה לבדוק אם קיים j עבורו $S[i] - S[j-1] = k$. הבעיה הומרה ל$2SUM$ קלאסית שכמובן נפתרת ב$O(n)$, $O(1)$ לכל איבר עם חיפוש בהאש.

2010 מועד א' אלגו שאלה 6

ארנב נמצא במשבצת $(1,1)$ של מערך משבצות בגודל $n\times n$. מטרתו להגיע לשק גזרים ב(n,n). מותר לו ללכת את הצעדים הבאים:

א. קפיצה מאוזנת למשבצת $(i,j+1)$ המותרת עבור $j<n$ וכן $i\leq n$

ב. קפיצה מאונכת למשבצת $(i+1,j)$, המותרת עבור $i<n, j\leq n$

ג. קפיצה אלכסונית $(i+1,j+1)$ המותרת עבור $i,j<n$.

כל צעד ממשבצת מסויימת מלווה בתשלום. על הארנב לשלם $p_1(i,j),p_2(i,j),p_3(i,j)$ עבור הצעדים המתוארים מעלה. זה ערך שנתון עבורכם לשאלה. נסמן ב$C(i,j)$ את העלות הקטנה ביותר עבור $1\leq i,j\leq n$ למעבר הארנב מהמשבצת $(1,1)$ אל (i,j)

כתוב אלגוריתם תכנון דינמי למציאת $C(n,n)$.

פתרון:

נרצה להמיר מעט את תנאי השאלה. כיוון שנרצה להתחיל ב$(1,1)$. נראה כי התנאי ב1 הוא ללכת ימינה , לכן אם אני ב(i,j) הגעתי מ$(i,j-1)$.

ב' הוא לרדת למטה, ולכן אם אני ב(i,j) הייתי ב$(i-1,j)$.

ג' זה לרדת ימינה למטה לכן אם אני ב(i,j) הייתי ב$(i-1,j-1)$.

כעת נכתוב את הנוסחה הבאה:

$$C(i,j):=\left\{\begin{array}{cc} min\{p_1(1,1),p_2(1,1),p_3(1,1)\} & i=j=1 \\ \infty & i>n\vee j>n \\ c(i,j-1)+p_1(i,j) & i=n\wedge j<n \\ c(i-1,j)+p_2(i,j) & j=n\wedge i<n \\ min\{c(i-1,j)+p_2(i,j),c(i,j-1)+p_1(i,j),c(i-1,j-1)+p_3(i,j)\} & o.w \end{array}\right\}$$

נכונות: נראה כי אם $i=j=1$ אזי אנחנו בתא הראשון ויתכנו שלושה ערכים שנרצה מהם בוודאות את המינימלי. אחרת, אם $i,j>n$ אנחנו כבר איכשהו חרגנו מהמטריצה ולכן פורמלית כאן נכתוב אנסוף כי אנחנו מנסים למזער את הסכום, אבל בפועל זה נועד למנוע זליגת זכרון ויציאה מתאי מטריצה. אם $i=n\wedge j<n$ אזי אנחנו בשורה האחרונה אך עדיין לא בעמודה האחרונה ולכן נזוז ימינה בלבד. אם $j=n\wedge i<n$ אנחנו בעמודה האחרונה ולכן נרצה רק לרדת למטה. אחרת, אנחנו במקרה בו ניתן לזוז כרצוננו במטריצה ולכן נבחור מינימום משלושת האפשרויות.

תכנון דינמי: ניצור מטריצה $n\times n$ ונקרא לה B. נאחסן $B[i,j]=c(i,j)$. נראה כי בהינתן שאני בתא מסויים אני זקוק לזה שמשמאלי, מעליי ושמאל למעלה ולכן נמלא בעמודות לאחר מילוי תנאי הבסיס. סה"כ מילוי תא יהיה $O(1)$ וכן אני זקוק למלא n^2 תאים ולכן סיבוכיות הזמן תהיה $O(n^2)$. כיון שאין צורך לשחזר פתרון נוכל לשמור בכל שלב שתי עמודות - נוכחית וקודמת וכך סיבוכיות המקום תרד ל$O(n)$. סה"כ הפתרון יהיה ב$B[n,n]$.

אלגו' 1 קיץ 2019 **שאלה 5 תרגיל בית**

בהינתן מס' n נרצה לדעת כמה עצים בינאריים שונים קיימים בעלי בדיוק n קודקודים. שימו לב כי עבור $n=2$ יש שתי אפשרויות, שורש עם בן שמאלי ושורש עם בן ימני. קלט: מס n ופלט: מס' עצים בינאריים מגודל n. פתור בתכנון דינמי

פתרון:

נרצה להסתכל על הבעיה כבעיית סכום ולא מקסום. נגדיר $f(i)$ כמה עצים בינאריים יש עם i קודקודים. עבור $i=1$, זה עץ יחיד. כלומר $f(1)=1$. עבור $i=2$ ממש נאמר לנו - שורש עם בן ימני או שמאלי כלומר $f(2)=2$. מתחיל להזכיר פיבונאצ'י....

אבל לצערנו זה לא יהיה.

נראה כי בהינתן n קודקודים, נקבע אחד להיות השורש, ואז נותר לסדר $n-1$ קודקודים בעץ. חלקם יהיו בתת העץ השמאלי, וחלקם בימניץ נגדיר $f(i)$ כך

$$f(i)=\left\{\begin{array}{cc} 1 & i=0 \\ 1 & i=1 \\ \sum_{k=0}^{i-1}f(k)f(i-k-1) & i\geq 2 \end{array}\right\}$$

מדוע? נסכום מס' אפשרויות לחלק מס' קודקודים $n-1$ לשני העצים. בהינתן שבחרתי k לשמאל, נותרו לי $i-k-1$ לימין (פחות אחד כי בחרתי גם שורש)

כעת לתכנון הדינמי = נרצה לדעת מראש ערכים קודמים ונשתמש במערך $B[i]=f(i)$. סה"כ זמן ריצה $O(n^2)$ ומקום $O(n)$.

תל אביב $2017BA$ **שאלה 1**

בהינתן k מערכים ממוינים בגודל n תאר אלגוריתם שפלטו מערך ממויין בגודל kn המכיל איברים של k המערכים הממויינים. זמן ריצה $O(nklogk)$

פתרון: נבנה ערימה בגודל k שיעלה $O(k)$. נכניס בתחילה את המינימום במערכים השונים, ונזכור עם פוינטר מאיזה מערך האיבר הגיע. נבצע הוצאה של איבר המינימום מהערימה ונכניס במקום את האיבר הבא במערך המותאם לאיבר שכעת הוצאנו. כך נבצע nk פעמים. להכניס ולהוציא יעלה $O(logk)$ ולכן סה"כ $O(nklogk)$.

סעיף ב -

נתונים k מערכים $A_1,..A_k$ לא בהכרח ממויינים עם n ערכים כל אחד. כל הערכים שונים. הצע מימוש למבנה הנתונים הבא:

א. $init(A_1,....,A_k)$ מאתחל מבנה באמצעות k מערכין. זמן ריצה $O(nk)$ בתוחלת ו$O(nk)$ מקום

ב. $whereIsValueI(i,D)$ - בהינתן ערך i (נתון ערך ולא מצביע לאיבר), יש להוציא כפלט את האינדקס של מערך שמכיל את הערך i אם הוא קיים באחד המערכים ו$null$ אחרת. זמן ריצה $O(1)$ במקרה הגרוע.

אסור להוסיף מערכים חדשים למבנה נתונים מאותחל.

פתרון:

נראה כי קיבלנו nk איברים. נרצה לבנות טבלת האש כי אמרו תוחלת וכן זה יעלה בדיוק $O(nk)$ מקום. נרצה בהינתן איבר להיות מסוגלים לגשת לאינדקס של המערך המקורי שלו. לכן כאשר נאתחל את המבנה, נכניס את כל האיברים עם מפתח הערך שלהם וכן נכניס להם שדה אינדקס-י מאיפה הם הגיעו. כעת בהינתן i נחפשו במבנה הנתונים ב$O(1)$, ואז נמצא את האיבר ונחזיר את השדה שלו.

תל אביב $2017BA$ **שאלה 3**

בכל רגע בקבוצה S שכעת תבנו לה מבנה נתונים יש לכל היותר איבר אחד שערכו i.

א. $init$ אתחל את s לקבוצה הריקה בזמן $O(1)$

ב. $insert$ מוסיף איבר x אל s

ג. $delete$ מוחק את x מs. נניח כי יש לנו מצביע לx ושx אכן נמצא בs

ד. $everDelete(i,s)$ מחזיר כן אם נעשתה בעבר פעולת מחיקה על איבר שערכו i. בין אם לאחר מחיקתו הוכנס מחדש לS איבר שערכו i או לא) אחרת מחזיר לא. בפעולה זו נתון ערך i ולא מצביע. יש לשים לב שאם אי פעם בתולדות המבנה נמחק איבר כזה יש להחזיר כן.

1. תאר מבנה נתונים שתומך ב$insert, delete$ ב$O(logn)$ ובפעולה ד' בזמן $O(logk)$ כאשר k הוא מס' פעולות המחיקה שבוצעו מאז שאותחל מבנה הנתונים. k הוא מס' פעולות ה$delete$ שבוצעו ויתכן שהוא מקיים $k << n$.

פתרון:

נשתמש בעץ AVL ראשי לניהול הקלט, אתחולו יעלה $O(1)$ וכן מחיקה והכנסה ב$O(logn)$.

כעת נשתמש בעץ עזר T' שהוא יהיה בגודל לכל היותר k. בעת פעולת מחיקה, אנחנו נסתכל על הערך של האיבר שמחקנו ונחפש אותו בT'. יעלה לנו $O(logk)$. אם מצאנו לא נעשה דבר, אחרת נכניסו לעץ. מה שיעלה גם כן $O(logk)$. לכן כאשר נמחק מהעץ יתבצע $logn + logk < logn + logn = O(logn)$ כי log פונקציה עולה. כעת בסעיף ד' כאשר נרצה לדעת האם בוצעה עליו פעולת מחיקה אי פעם נלך לחפשו בעץ T' מה שיעלה $O(logk)$. מש"ל.

2. נוסיף למבנה את הפעולה הבאה:

$stillDeleted(i,s)$ - מחזיר כן אם לאחר המחיקה האחרונה של האיבר שערכו i מS לא הוכנס איבר חדש שערכו i. אחרת מחזיר לא. זמן ריצה כמו קודם לשאר הפעולות וכן לפעולה החדשה $O(logk)$

פתרון:

נוסיף שדה אל השדה T'. כאשר נכניס איבר אל העץ אנחנו נלך לפני לחפש אותו בT' מה שיעלה $O(logk)$. אם מצאנו אותו - אזי שהוא היה מחוק והוא כעת מוכנס מחדש. ולכן השדה שנכניס יקרא $isHere$. השדה יתחזק עבור כל האיברים שנמחקו אי פעם, מה הסטטוס הנוכחי שלהם בעץ. בעת ההכנסה, יאותחל שדה זה ל$true$. סה"כ ההכנסה תהיה ב$O(logn) + O(logk) = O(logn)$ כי שוב log פונקציה עולה ו$k << n$. המחיקה מהעץ לא תשתנה, רק שאם כעת מחקנו איבר נעדכן את ערך השדה שלו ל$false$. שוב, $O(logn)$. למימוש ד' כמו קודם ולמימוש הפעולה החדשה $O(logk)$ כי נלך לחפש בT' ונשלוף את הערך.

$2017BB$ תל אביב שאלה 1

נרצה מבנה נתונים שמתחזק שתי קבוצות מספרים S ותומך ב: אתחול לקבוצה הריקה של הקבוצות, הוספת מס' לכל אחת, מחיקת מס' מכל אחת בהינתן פוינטר, וכן:

מחזיר כן אם לA וB יש חיתוך לא ריק - כלומר יש מס' שנמצא בשניהם.

בכל הסעיפים $n = |A| + |B|$.

א. תאר מבנה התומך בכל הפעולות בזמן $O(logn)$

פתרון: עץ AVL וקיבלנו הכל פרט לפעולה האחרונה.

נתחזק 3 עצי AVL: אחד לאיברי A, אחד לאיברי B ואחד לכפולים.

בעת כניסת איבר לקבוצה A, נבדוק אם הוא קיים בקבוצה B. אם כן? נכניס אותו אל עץ הכפולים. באופן דומה בעת כניסה לאיבר B. כאשר נמחק איבר, נבדוק האם הוא קיים גם בקבוצה השניה, אם כן נמחק אותו מהקבוצה שלו וכן מעץ הכפולים - כי אין כפילות עוד. כאשר נרצה לבדוק האם קיים איבר בשני העצים - נבצע חיפוש על עץ הכפולים ב$O(logn)$

ב. תאר מבנה נתונים עבורו זמן הריצה של כל הפעולות מסעיף א' הוא $O(1)$ בתוחלת. יתבצעו לכל היותר N פעולות על מבנה הנתונים כאשר N ידוע למבנה מראש. כמו כן ניתן להניח שמותר לאתחל מערך ריק בגודל N בזמן $O(1)$

פתרון: שני רמזים חשובים = לבנות מערך בגודל מס' הפעולות ו2 הוא המילה תוחלת.

נכניס את האיברים לטבלה אחת לפי הערך שלהם. לכל היותר ייתכנו בתוך טבלת האש שלנו 2 איברים שמתאימים לכל ערך - אם יש לו איבר מקבוצה A ואיבר מקבוצה B. כלומר בעת ההכנסה נצמיד לאיבר שדה שיציין מהיכן הוא הגיע. אם נרצה להכניס איבר שלא היה קיים עוד לפני בטבלה, נכניס אותו ממש עם השדה. אם נכניס איבר מB שכבר קיים בטבלה עבור A נעדכן את השדה שגם B בפנים. כאן הכנסה מחיקה בקלות. האם יש מס' שנמצא בשניהם? מה שנעשה יהיה לעדכן קאונטר של מס' האיברים השונים בטבלה. כמו כן נעדכן קאונטר של מס' האיברים שכעת כפולים כלומר קיים להם עותק מA ועותק מB. אם הקאונטר שווה אפס אזי הקבוצות זרות.

מבחן 2024 קיץ ב'

שאלה 1:

נתונים שני מערכים ממויינים $A[1,..\frac{n}{2}]$ ו$B[1,...\frac{n}{2}]$. בגודל זהה כמתואר. הניחו שבכל מערך האיברים שונים זה מזה ואין למערכים איברים משותפים.

א. כתוב אלגוריתם רקורסיבי למציאת החציון במערך המאוחד של A וB בזמן $O(logn)$

פתרון:

תמיד שנרצה רקורסיבי נזכר בנוסחת הנסיגה $T(n)=T(\frac{n}{2})+O(1)=O(logn)$ כאשר n הוא מס' האיברים המשותף במערך המאוחד הכולל. המטרה כאן היא כמובן שלא לאחד את המערכים כי אז נפעיל סלקט ב$O(n)$. נראה כי נרצה להפטר מחצי מהאיברים בכל שלב. נסתכל על איברי החציון של A וB (ממויינים ולכן אנחנו יודעים מיהם) כעת נסמנם בהתאמה $A[\frac{n}{4}]$ ו$B[\frac{n}{4}]$.

אם $A[\frac{n}{4}]>B[\frac{n}{4}]$ זה אומר שהחציון של A גדול מהחציון של B, ולכן האיברים בהם נלך לחפש כאן יהיו החצי האחרונים של B עם החצי הראשונים של A שהרי כעת שם יש $\frac{n}{2}$ איברים ובהם פוטנציאל להיות החציון

אם $A[\frac{n}{4}]<B[\frac{n}{4}]$ כעת החציון של B גדול מהחציון של A ולכן כעת נלך לחפש בחצי הראשונים של B עם חצי האחרונים של A.

מדובר בהפרד ומשול קלאסי.

ב. הוכיחו את סיבוכיות זמן הריצה של הפסודו/ אלגוריתם שהצגתם בסעיף הקודם

פתרון:

לא ברור איך התכוונו לכן נציג שתי שיטות.

א. מאסטר קלאסי - $a=1,b=2,log_2 1=0$ ולכן $n^0=O(1)$ ולפי מקרה 2 אזי $O(logn)$

ב. באינדוקציה.

בסיס: $n=1$ נקבל כמובן $T(1)\in O(logn)$

צעד: נניח נכונות לכל $n'<n$ כלומר $T(\frac{n}{2})\leq clog(\frac{n}{2})$ ולכן נקבל

$$T(n)=T(\frac{n}{2})+1\leq clog(\frac{n}{2})+1=clogn-c+1\leq clogn$$

המעבר יהיה נכון אמ"מ $1-c\leq 0$ כלומר לכל $c\geq 1$ ו$n=1$. כנדרש

ג. כתוב אלגוריתם למציאת האיבר הk בגודלו במערך המאוחד של A וB כאשר $1\leq k\leq \frac{n}{2}$ זמן ריצה נדרש $O(logk)$

פתרון:

האיבר הk בגודלו חייב להמצא בk האיברים הראשונים של A או B. לכן נסתכל עליהם בלבד. גודל הקלט ההתחלתי שלנו יהיה $2k$ ונרצה לעבוד בדומה לסעיף א' לנסות לצמצם את גודל הקלט. אם $k=\frac{n}{2}$ זה בדיוק הסעיף הראשון. כעת נסתכל בכל שלב על החציונים בk האיברים הראשונים בכל מערך. לפני כן נחלק למקרים.

נסתכל על האיבר המקסימלי במערך A והמינימלי במערך B. אם המקסימלי קטן מהמינימלי אזי שכל איברי מערך אחד קטנים מהשני. כעת, החיפוש יתבצע במערך הראשון בלבד. באופן סימטרי על המערך השני.

אחרת אנחנו לא במקרה קצה שכזה, לכן נסתכל על החציונים. אם סכום האיברים עם החציונים משמאל בשני המערכים קטן שווה לk נלך רקורסיבית שמאלה. אחרת, הוא גדול מk לכן נרצה ללכת ימינה. סה"כ נקבל

$$T(2k)=T(k)+O(1)=O(logk)$$

שאלה 2: הצע מבנה נתונים S שמתחזק n מפתחות שונים זה מזה. הנח שקיימת פונקציה $Time(S)$ שמחזירה ב$O(1)$ את הזמן שעבר מאתחול המבנה S. אין צורך לממש אותה

א. חיפוש מפתח k בזמן $O(logn)$

ב. הכנסה מפתח k למבנה בזמן $O(logn)$

ג. מחיקת מפתח k בזמן $O(logn)$

ד. $MaxTimeGap(S)$ מחזיר את המפתח k במבנה עבורו ההפרש בין זמן הכנסת k לבין הכנסת הקודם למפתח k הוא מקסימלי. זמן ריצה $O(1)$.

פתרון:

נשתמש כמובן בעץ AVL. נקבל את א+ב+ג בחינם. כעת האתגר הוא ד'.

נוסיף לכל איבר בעת הכניסה את השדה $time$. כעת בעת הכניסה למבנה נחפש את הקודם של מי שהנכנסנו:

מציאת קודם בעץ -

אם יש לצומת תת עץ שמאלי, הקודם יהיה הכי גדול בתת העץ השמאלי. (בתנאי שאכן ערך המפתח שלו הוא אחד פחות)

אם אין, טפס למעלה בעץ כל עוד אתה בן שמאלי ותחזיר הורה ראשון שאתה בן ימני שלו. (וגם הערך שלו הוא אחד פחות יענו $k-1$)

כמובן שאם לא נמצא קודם אז נגדיר אפס כי השאלה לא התייחסה בבירור מה קורה למי שאין עוקב. חיפוש עוקב עולה $O(logn)$ ולא משפיע אסימפטוטית. כעת, לכל איבר נשמור בשדה נוסף את ההפרש בין זמן ההכנסה שלו לזמן ההכנסה של הקודם לו, אם כמובן יש אחרת נגדיר אפס. עד כאן לא השפענו אסימפטוטית.

כעת אנחנו יודעים לכל איבר מה זמן ההפרש, אך לא יודעים את המקסימלי ולכן נוסיף שדה נוסף שיהיה קל לתחזוק: לכל תת עץ נגדיר מיהו הk עם ההפרש המקסימלי, כמובן שנוכל לתחזק זאת בזמן ההכנסה וההוצאה מבלי להשפיע אסימפטוטית שכן ראינו שניתן להוסיף שדות כאלו "בחינם" אסימפטוטית. כעת, בפעולה ד' פשוט נחזיר את הערך שמתאים לשדה של הראש וסיימנו.

שאלה 3:

הגדרה: יהי p קודקוד בעץ. העומק של p הוא אורך המסלול מהשורש לקודקוד p. עומק השורש הוא אפס. נתון עץ בינארי סופי המכיל m עלים. נסמן ב$d_1,....d_m$ את עומק העלים.

א. הוכח באינדוקציה על מבנה העץ: $\sum_{i=1}^m 2^{-d_i} \leq 1$ (15 נק')

פתרון:

נוכיח באינדוקציה על מס' העלים. $m=1$ נקבל יש שורש אחד בלבד שהוא עלה ומתקיים אכן כי $d_1=0$ ואכן $2^{-0}=1\leq 1$.

צעד: נניח נכונות לכל עץ בינארי עם $m'<m$ עלים. נסתכל על בניו של השורש ונסמנם x_1,x_2. בהכרח מס' העלים בכל אחד מהם קטן מm. ולכן ניתן להפעיל עליהם את הנחת האינדוקציה.

נסמן את העלים בעץ x_1 כ$m_1,......m_k$ ובדומה בx_2 נסמן $a_1,......a_y$. נשים לב כי

$$\sum_{i=1}^{y} 2^{-d_i} \leq 1, \sum_{i=1}^{k} 2^{-d_i} \leq 1$$

כעת נראה כי מס העלים הכולל בעץ הוא מס' העלים בשני תת העצים כלומר $y+k$. עם זאת, עומקם אינו שהיה אלא גבוה באחד. כלומר עומק כל אחד מהם יהיה d_i+1. מכאן נקבל

$$\sum_{i=1}^{y+k} 2^{-d_i-1} = \frac{1}{2}\sum_{i=1}^{y+k} 2^{-d_i} = \frac{1}{2}\sum_{i=1}^{y} 2^{-d_i} + \frac{1}{2}\sum_{i=1}^{k} 2^{-d_i} \leq \frac{1}{2}*1+\frac{1}{2}*1 = 1$$

כאשר המעבר $\leq$נבע מהנחת האינדוקציה. כנדרש.

ב. הגדרה: יהי T עץ בינארי. נאמר כי T מלא אם לכל קודקוד פנימי יש שני בנים. הוכיחו כי בעץ בינארי מלא מתקיים $\sum_{i=1}^m 2^{-d_i}=1$.

הוכחה: שוב באינדוקציה על מס' העלים. כעת,

בסיס: $m=1$ נקבל כמו קודם שורש יחיד ואכן $2^{-0}=1$ ממש.

צעד: נניח נכונות לכל עץ בינארי מלא עם $m'<m$ עלים. נסתכל על בניו של השורש ונסמנם x_1,x_2. בהכרח מס' העלים בכל אחד מהם קטן מm. כמו כן כל אחד מהם הוא בהכרח עץ בינארי מלא. ולכן ניתן להפעיל עליהם את הנחת האינדוקציה.

נסמן את העלים בעץ x_1 כ$m_1,......m_k$ ובדומה בx_2 נסמן $a_1,......a_y$. נשים לב כי

$$\sum_{i=1}^{y} 2^{-d_i} = 1, \sum_{i=1}^{k} 2^{-d_i} = 1$$

כעת נראה כי מס העלים הכולל בעץ הוא מס' העלים בשני תת העצים כלומר $y+k$. עם זאת, עומקם אינו שהיה אלא גבוה באחד. כלומר עומק כל אחד מהם יהיה d_i+1. מכאן נקבל

$$\sum_{i=1}^{y+k} 2^{-d_i-1} = \frac{1}{2}\sum_{i=1}^{y+k} 2^{-d_i} = \frac{1}{2}\sum_{i=1}^{y} 2^{-d_i} + \frac{1}{2}\sum_{i=1}^{k} 2^{-d_i} = \frac{1}{2} * 1 + \frac{1}{2} * 1 = 1$$

שאלה 4:

יהי $A[0,....k]$ מערך ביטים שמייצג מונה בינארי. הנח כי המונה מאותחל להיות 0. המונה תומך בפעולתץ $incerement(A)$ שמגדיל ערך המונה ב1. לכל $1 \leq i \leq k$ עלות להיפוך הביט $A[i]$ היא 2^i (בשונה מהעלות הקבועה שראינו בכיתה בעלות לשיעורין).

נדון בעלות סדרה של n פעולות הגדלה כאשר $n < 2^{k+1}$.

א. בהינתן n פעולות העלאה, מה עלות פעולת העלאה במקרה הגרוע כפונקציה של n? הוכיחו תשובתכם

פתרון:

המקרה הכי גרוע הוא מצב בו יש לנו רצף של אחדות. במצב זה כל פעולת העלאה תחליף לנו k ביטים. נצטרך להחליף את כל הביטים וזה יעלה לנו סה"כ

$$2^0 + 2^1 + ... + 2^{log_2 n} = \sum_{i=1}^{logn} 2^i = \frac{1(2^{log_2 n+1}-1)}{1} = 2n-1 = O(n)$$

ב. בהינתן n פעולות העלאה, מה העלות לשיעורין לפעולת העלאה אחת? הוכיחו תשובתכם.

שוב נסתכל על המקרה הגרוע. היפוך ביט הראשון יתרחש בערך n פעמים. הבא אחריו $\frac{n}{2}$ פעמים. הבא אחריו $\frac{n}{4}$ פעמים ובמקרה הכללי כ$\frac{n}{2^k}$פעמים. ולכן סה"כ נקבל אם נכפיל בעלות ההיפוך

$$\sum_{k=1}^{logn} \frac{n}{2^k} * 2^k = \sum_{k=1}^{logn} n = nlogn$$

ולכן העלות לשיעורין עבור פעולה בודדת יהיה $logn = \frac{nlogn}{n} \leq \frac{T(n)}{n}$ כלומר $O(logn)$ לשיעורין.

2025 מועד ג'

שאלה 1:

נגדיר ערמה d להיות עץ שלם (כל העלים באותה רמה, ובפרט לכל אחד שני בנים) בו לכל צומת יש d ילדים וכל אב גדול מכל בניו (כמו בערימת מקסימום). תאר כיצד ניתן לאחסן ערימה d במערך (כמו ערימ בינארית) ותאר כיצד מחשבים את האינדקס של ההורה של צומת במיקום i והילדים של צומת במיקום i.

ב. תאר את הפעולות הבאות על ערימהd, הכנסה והוצאת מקסימום. נתח סיבוכיות של כל פעולה כפונקציה של n וd.

פתרון:

בדומה לערימה בינארית, שם אם אב בצומת i בנים בצומת $2i+1$ ו$2i+2$ והאב היה במיקום $\frac{i-1}{2}$ (בערך שלם תחתון)

באופן דומה אם אבא בצומת i בניו יהיו בצומת $di+1, di+2, ...di+d$. כמו כן למציאת האבא של צומת במיקום i נבצע $\frac{i-1}{d}$בערך שלם תחתון.

כעת נעבור למימוש הפעולות:

הכנסה לערימה רגילה מתבצעת עם הכנסת למיקום האחרון ביותר וכך פעפוע כלפי מעלה בגובה העץ.

גם כאן אנחנו נכניס לערימה במיקום $A[n]$, כך שמרנו על הערימה כעץ בינארי שלם, וכעת נרצה לפעפע את האיבר כלפי מעלה בשביל לשמור על איזון. נראה כי הכנסתו במיקום זה יעלה $O(1)$ והפעפוע יעלה כגובה העץ. כלומר סה"כ $O(h)$, כעת נצטרך לשאול עצמנו מהו גובה העץ? העץ שלם ולכן בהינתן x עלים יש $x-1$ קודקודים פנימיים, נראה כי ברמה האפס יש $d^0=1$ קודקודים, ברמה הראשונה ישנם d ילדים, לאחר מכן יש d^2 ילדים וכן הלאה ברמה הi יש d^i ילדים. כמו כן, ברמה האחרונה יהיו d^h קודקודים כי הרמה האחרונה בגובה h. לכן סה"כ מס' הקודקודים בעץ יהיה

$$1+d+d^2+.....+d^h=\sum_{k=0}^{h}d^k=\frac{1(d^h-1)}{d-1}$$

כעת, אמרנו כי ברמה האחרונה d^h קודקודים ולכן זה מס' העלים, לכן מס' הקודקודים הפנימיים יהיו d^h-1. סה"כ מס' הקודקודים בעץ יהיה $2d^h-1$. כלומר $n=2d^h-1$ ולכן אם נעביר נקבל $n+1=2d^h$ ונחלק בשתיים $\frac{n+1}{2}=d^h$ ולכן אם נפעיל לוג נקבל $h=log_d(\frac{n+1}{2})=O(log_dn)$, וסה"כ גובה העץ יהיה $O(log_dn)$, לכן סיבוכיות ההכנסה תהיה $O(log_dn)$

כעת באשר להוצאת המקסימום, זה יהיה פשוט האיבר $A[0]$, כעת נוציא אותו מהערימה, נשים במקומו את האיבר $A[n]$ ונפעפע כגובה העץ ב$O(log_dn)$ כפי שראינו קודם.

שאלה 2:

א. נגדיר עץ מושרש עם עלים בגובה קבוע להיות עץ בינארי בו כל העלים נמצאים בדיוק באותו העומק. הוכיחו/הפריכו: יהי T עץ מושרש עם עלים בגובה קבוע עם n קודקודים, אזי גובה העץ הוא לכל היותר $logn$.

פתרון: למעשה זו ההגדרה של עץ בינארי שלם, עם סוויץ, נשים לב כי בכל רמה מס' הקודקודים יהיה לכל היותר 2^h, ולכן סה"כ מס' הקודדודים בעץ יהיה

$$1+2+4+.....+2^h=\frac{1(2^h-1)}{2-1}=2^h-1$$

זה עדיין לא אומר לנו כלום. אם נקח שרוך בגודל 4, שכל הזמן הולכים הצידה, יש ערך אחד שהוא עלה ולכן על העלים באותה רמה, אך גובה עץ שכזה הוא 3! והרי ש$log4=2$, ולכן הטענה שגויה.

ב. הוכח באינדוקציה על מבנה העץ שגובה עץ פיבונאצ'י עם n צמתים הוא $O(logn)$.

בסיס: $n=1$ אזי קודקוד יחיד ואכן גובה עץ שכזה הוא אפס ומתקיים $log1=0$

$n=2$ מתקיים שיש קודקוד עם בן שמאלי, גובה עץ שכזה הוא 1 ואכן $log2=1$

צעד: נניח נכונות לכל עץ עם $n'<n$ קודקודים. יהי עץ פיבונאצ'י עם n קודקודים. נניח שהעץ הוא מסדר i. אזי יש לנו שני עצים - עץ ימני F_{i-1} ותת עץ שמאלי שהוא בנו והוא F_{i-2}. שניהם מקיימים הנחת האינדוקציה כי בכל אחד מהם יש פחות מn קודקודים. נסמן את מס' הקודקודים בעץ F_{i-1} כn_1, ובהתאמה n_2. מתקיים $n=n_1+n_2+1$. כעת, גובה כל אחד מהעצים חסום בהתאמה על $log(n_1),log(n_2)$. כעת,

$$h(T)=1+max\{h(T_{i-1}),h(T_{i-2})\}=1+max\{log(n_1),log(n_2)\}\leq 1+log(n_1)\leq 1+log(n)=O(logn)$$

כאשר בה"כ הנחנו $n_1>n_2$ וlog פונקציה עולה. באופן דומה יכולנו לומר $n_2>n_1$ וההוכחה הייתה זהה.

שאלה 3:

להלן הגדרת הבעייה "בעיית החלוקה חצי חצי"

קלט: מערך A של מספרים ממשיים שונים זה מזה בגודל n זוגי

פלט: תמורה של המערך A כך ש$\frac{n}{2}$ המספרים הקטנים ביותר ממוקמים בצד השמאלי, ו$\frac{n}{2}$ המספרים הגדולים ביותר בצד הימני. סדר ימני בתוך כל אינדקס אינו נדרש.

א. מה החסם התחתון על מס' ההשוואות במקרה הגרוע של כל אלגוריתם מבוסס השוואות הפותר את בעיית החלוקה חצי חצי לקלט על מערך בגודל n?

ניתן להעזר בנוסחת סטרלינג - $n!\approx\sqrt{2\pi n}(\frac{n}{e})^n$

ב. הצע אלגוריתם יעיל לפתרון בעיית החלוקה חצי חצי.

פתרון:

ראשית נעשה את סעיף ב'. האלגוריתם יהיה פשוט מאוד: נפעיל סלקט על איבר החציון, לאחר מכן כל האיברים שקטנים ממנו יהיו משמאלו וכל אלו שגדולים ממנו מימינו. סה"כ $O(n)$ עבודה. כעת קיבלנו כיוון לסעיף א', להוכיח כי אולי (לא בטוח) שהחסם התחתון יהיה n.

כאשר הוכחנו חסם תחתון הראינו שמס' הפרמוטציות האפשריות הוא $n!$. אך זה לא מה שרלוונטי עבורי כרגע, כי אינני מעוניין למיין. אני רק רוצה לבחור חצי מהאיברים ולמקם אותם משמאל למשל והשאר יהיו מימין, כלומר אנחנו מדברים על $\binom{n}{\frac{n}{2}}$. ננסה להבין מה סדר גודל הביטוי

$$\binom{n}{\frac{n}{2}} = \frac{n!}{(\frac{n}{2})! * (\frac{n}{2})!} \approx \frac{\sqrt{2\pi n}(\frac{n}{e})^n}{(\sqrt{2\pi\frac{n}{2}}(\frac{\frac{n}{2}}{e})^{\frac{n}{2}})^2} = \frac{\sqrt{2\pi n}(\frac{n}{e})^n}{\pi n(\frac{n}{2e})^n} = \frac{\sqrt{2\pi n}}{\pi n}(\frac{\frac{n}{e}}{\frac{n}{2e}})^n = \sqrt{\frac{2\pi n}{\pi^2 n^2}}(2)^n$$

$$= \sqrt{\frac{2}{\pi n}} * 2^n \leq 2^n$$

כלומר, מס' העלים יהיו חסומים ב2^n, ולכן מס' הקודקודים הפנימיים יהיה חסום ב$2^n - 1$, נקבל שמס' הקודקודים חסום ב$2^{n+1} - 1 < 2^{n+1}$, נפעיל לוג ונקבל $h \leq log_2(2^{n+1}) = n + 1$, כלומר חסם תחתון לביצוע המשימה יהיה $\Omega(n)$.

2020 אלגו 1 שאלה 5 מועד א

נרצה לרכוש מוצר בחנות ונתונות לנו n סוגי מטבעות שונים שמיוצגים ע"י המחירים $v_1, ..., v_n$. נרצה להשתמש במס' המטבעות הקטן ביותר. כמו כן נתון $1 = v_1 < v_2 < .... < v_n$

ניתן להשתמש בכל מטבע פעם אחת.

קלט: מערך מטבעות ועלות מוצר k.

פלט: מס' מטבעות כך שכמות המטבעות לתשלום אופטימלית.

פתרון:

נגדיר $f(i, x)$ כפונקציה שמחזירה את מס' מטבעות מינימלי עבור המטבעות $1, ...i$ עם עלות מוצר x. כמובן $1 \leq i \leq n$ וכן $0 \leq x \leq k$. הפתרון יהיה $f(n, k)$.

נשים לב שבכל שלב בפנינו מס' אפשרויות:

א. אם שווי המטבע הi גדול מהערך שנותר לנו, לא נוכל להשתמש בו.

ב. אחרת, נבחר את הmin מבין להשתמש במטבע, או לא להשתמש במטבע. אם לא השתמשנו במטבע - נרד לתת בעיה $f(i-1, x)$. אם השתמשנו במטבע נרד לתת בעיה $f(i, x - v_i)$ כי יתכן שנרצה להשתמש בו שוב בהמשך.

כמו כן, אם $x = 0$ אזי לא צריך להגיע לאף סכום ולכן מס' המטבעות המינימלי יהיה אפס. אם $i = 0$ וכן $x > 0$ אזי שלא ניתן להגיע לסכום! לכן נגדיר שם אנסוף גם כן כהנחה שניקח כרגע.

נגדיר את הנוסחה הבאה:

$$f(i, x) := \left\{ \begin{array}{cc} 0 & x = 0 \\ \infty & i = 0 \wedge x > 0 \\ f(i-1, x) & v_i > x \\ min\{f(i, x - v_i) + 1, f(i-1, x)\} & o.w \end{array} \right\}$$

ניצור מטריצה בגודל $n \times k$. נקרא לה A. $A[i, x] = f(i, x)$. נמלאה ראשית עבור תנאי בסיס, ולאחר מכן נראה כי בהינתן שאני בתא מסויים אני זקוק למי שמשמאלי או מעלי ולכן נמלא בעמודות משמאל לימין. סה"כ מילוי תא יעלה $O(1)$ ונשתמש בnk תאים ולכן סיבוכיות הזמן תהיה $O(nk)$. נראה כי גם ניתן למלא בשורות, ולכן נבחר בעת האתחול במי אנחנו מעוניינים למלא, ונשמור שתי שורות קודמות/ שתי עמודות קודמות וסיבוכיות המקום תהיה $O(min\{n, k\})$.

2016 תל אביב שאלה 1

א. נתונים n זוגות של מפתחות (a_i, b_i). לכל מפתח $1 \leq i \leq n$ מתקיים $a_i, b_i \in [1, 10n]$. תאר מבנה נתונים לבנייה בזמן $O(n)$ שתומך בפעולה הבאה בזמן $O(1)$- $searchAll(a)$ מחזיר רשימה של כל זוגות המפתחות בהם a הוא המפתח השמאלי בהם. אורך הרשימה יכול להיות בין 0 לn.

פתרון:

ראשית נמיין את המפתחות לפי ערך a. נוכל לבצע זאת באמצעות מיון מניה (תחום המספרים נתון) ב$O(n)$. כעת בידינו הזוגות באופן ממויין לפי a. נבנה מערך בגודל $10n$ שהרי נתון שהמספרים שלמים. כל איבר עם ערך a יכנס למיקום $A[a]$. נראה כי יתכנו כמה שמתאימים לאותו ערך a ולכן כל מערך יכיל מצביע לרשימה מקושרת. בעת הבנייה, כאשר נגיע ל$A[a]$ ונראה שיש שם איבר נוסיף לו ברשימה המקושרת איבר נוסף בסופה. סה"כ זה יעלה זמן לינארי. כעת, בהינתן a נתון נלך אל $A[a]$ ונחזיר למי שקרא לשאילתה מצביע לרשימה.

ב. הפעם נתון שכל המפתחות שלמים מהתחום $[1, n^3]$. תאר מבנה נתונים שניתן לבנותו בזמן $O(n)$ במקרה הגרוע, ותומך בפעולה $SearchAll$ בזמן $O(logn)$. אסור לבנות את המבנה בזמן גדול מ$O(n)$ ומקום.

פתרון:

כעת לא נוכל לבנות מערך ענק שכזה כי זה יקח מלא מקום. מה כן? נמיין הפעם עם מיון בסיס ע"י בחירת בסיס n ונקבל $d = log_n n^3 = 3$ ואז $O(3(n+n)) = O(6n) = O(n)$. כלומר כעת מיינו אותם שוב לפי a בזמן $O(n)$. כעת נבנה עץ AVL ממערך ממויין. מה שנעשה יהיה ראשית לבנות עץ AVL שבתוכו ערכי a, ובתוך כל אחד יהיה רשימה מקושרת

לכל הערכים השונים של הקבוצות השונות (כמו קודם). כעת נבנה את העץ רקורסיבית כפי שלמדנו בהרצאה, וכל פעם שנתקל בערך שראינו נכניסו לרשימה מקושרת המתאימה. סה"כ נבנה כך ב$O(n)$. כעת לחיפוש נחפש את a בעץ $O(logn)$ ונחזיר את המצביע לרשימה המקושרת שלו. הערה - נעבור לינארית על המערך ונקבץ את הזוגות עם אותו ערך a לרשימה מקושרת, נקבל מערך A' עם ערכי a השונים ממויינים וכעת מהם נבנה עץ AVL.
דרך אלטרנטיבית - מערך של ערכי הa, שכל אחד יצביע כך לרשימה מקושרת. כעת נחפש בינארית ב$O(logn)$ ונגיע למצביע המתאים.

2016 תל אביב שאלה 2
תאר מבנה נתונים שאיבריו זוגות מפתחות (a, b) התומך בפעולות הבאות:
א. הכנס זוג (a, b) למבנה. הנח שלא קיימים כבר. $O(1)$ בתוחלת
ב. $SearchAll(a)$ החזר רשימה של זוגות המפתחות שa הוא המפתח השמאלי בהם. על הפעולה להתבצע בזמן $O(1)$ בתוחלת. מניחים שמס' המפתחות כולל לא יעלה על n ומותר אתחול בזמן $O(n)$ ובפרט אורך רשימת הפלט של $SearchAll$ יכול להיות בין 0 לn.

פתרון:
כעת זו אותהה שאלה כמו קודם, ללא המיון, עם המילה תוחלת. נרצה לבנות טבלת האש, בה המפתח יהיה האיבר a השמאלי בזוג. בנייתה מראש תעלה $O(n)$ כנדרש לאתחול. כעת, כאשר נכניס איבר לטבלה נכניס את ערך a שלו, וניצור לו מצביע לרשימה מקושרת (נשמור פוינטר לסוף), בה אנחנו נוסיף את הזוג (a, b) לרשימה. סה"כ הוספה לרשימה מקושרת בהינתן מצביע היא $O(1)$, כעת אם חיפשנו בטבלה את a והוא כבר קיים, נחפש כמובן ב$O(1)$ בתוחלת, נגיע לאיבר ונוסיף את (a, b) כאיבר חדש ברשימה המקושרת ונקדם את הפוינטר. סה"כ כשנרצה להחזיר רשימה נבצע חיפוש של a בטבלה, ונחזיר את הפוינטר לרשימה המקושרת.

סעיף ב - הוסף למבנה הנתונים את הפעולות הבאות:
א. $Delete(a, b)$, מוחק זוג מפתחות (a, b) אם קיים. $O(1)$ בתוחלת.
ב. $Search(a, b)$ החזר זוג מפתחות a, b אם קיים. אחרת $null$. $O(1)$ בתוחלת.
שנה את פעולת $insert$ כך שתכניס זוג למבנה רק אם אינו קיים בו. הפעולות כולן $O(1)$ בתוחלת.
פתרון:
כעת נשנה את מבנה הנתונים כי חיפוש ברשימה מקושרת עולה $O(n)$. ניצור טבלת האש בה ערכי a, וכל ערך a יצביע לטבלת האש נוספת.
חיפוש: נחפש לפי a, אם קיים יופי נתקדם לחיפוש בתוך b. אם קיים נחזיר זוג. סה"כ $O(1)+O(1)=O(1)$ לחיפוש בתוחלת.
הכנסה: ראשית נחפש, אם לא קיימים נכניסם. אם לא קיים a- נדרש להכניס את a לטבלה הגדולה, וליצור לו טבלת האש חדשה (אתחול $O(1)$), ונוסיף לשם את b. אם a כבר קיים פשוט נוסיף לטבלה שלו את b. סה"כ $O(1)$
מחיקה: נחפש את a, b. אם מצאנו, נבצע מחיקה של b ראשית מטבלת האש של a. כעת נעדכן כי נחזיק קאונטר לכל איבר כמה איברים יש בטבלת האש שלו. אם כעת הקאונטר יורה על אפס, נמחק גם את a מהטבלה הגדולה. סה"כ $O(1)$. אם קאונטר לא התאפס כמובן שנפסיק ונוריד קאונטר.
פעולה אחרונה- $searchAll$ נחפש a ונחזיר מצביע לטבלת האש של כל ערכי b. רשימת הזוגות תהיה (a, b') כאשר b' שייך לטבלת האש של a. סה"כ $O(1)$.

2024 קיץ ג'
שאלה 1:
עץ טרינארי הוא עץ בו לכל צומת לכל היותר שלושה בנים. עץ טרינארי כמעט שלם הוא עץ טרינארי בו כל המאות מלאות פרט אולי הקודקודים נמצאים בצורה רציפה מהקצה השמאלי ברמה התחתונה (כמו בבינארי), ערימת מקסימום טרינארית הוא עץ טרינארי כמעט שלם בו לכל צומת v מפתחות בניו קטנים ממפתח v.
א. כיצד ניתן לייצג באופן יעיל ערימת מקסימום טרינארית במערך?
פתרון:
באשר לצומת ברמה הi בניו יהיו בקודקודים $3i+1, 3i+2, 3i+3$ וכן אם נרצה להגיע לאבא של צומת ברמה הi נעשה $\frac{i-1}{3}$ בערך שלם תחתון.
ב.יהי A מערך המייצג ערימת מקסימום טרינארית, כתוב אלגוריתם לפעולות הבאות:
1. $Perent(A, i)$ - מקבלת ערימה ואינדקס i ומחזירה אינדקס לאב הצומת המיוצגת ב$A[i]$
פתרון: כפי שאמרתי מעלה, בשביל להגיע לאבא של הצומת נעשה $\frac{i-1}{3}$ בערך שלם תחתון ונגיע לאבא שלו ב$O(1)$ (אם חישוב אריתמטי מתבצע בזמן קבוע)
2. $Left(A, i)$ פונקציה מקבלת מערך A ואידנדקס i ומחזירה אינדקס של בן שמאלי של הצומת המיוצגת ב$A[i]$.
פתרון: נחזיר את $3i+1$ אם קיים, אחרת נחזיר שלא קיים.
3. כמו 2 רק בן אמצעי
פתרון: נחזיר את $3i+2$ אם קיים, אחרת נחזיר שלא קיים.

4. כמו 2 רק בן ימני
פתרון: נחזיר את $3i+3$ אם קיים, אחרת נחזיר שלא קיים. (הבהרה - לא קיים זה האינדקס לא מיוצג במערך).
ג. מה גובהו המדוייק (לא אסימפטוטי) של עץ טרינארי עם n צמתים?
פתרון:
ברמה הראשונה ישנו בן יחיד. בשנייה - 3 בנים, בשלישית 9 וסה"כ עד לרמה הלפני האחרונה נקבל

$$1+3+3^2+.....+3^{h-1}=\sum_{i=0}^{h-1}3^i=\frac{1(3^h-1)}{3-1}=\frac{3^h-1}{2}$$

מס' האיברים ברמה התחתונה הוא לכל היותר 3^hבמקרה הגרוע בו העץ שלם לחלוטין. במצב כזה מס' הקודקודים יהיה

$$1+3+3^2+.....+3^{h-1}+3^h=\sum_{i=0}^{h}3^i=\frac{1(3^{h+1}-1)}{3-1}=\frac{3^{h+1}-1}{2}$$

כלומר במצב בו הערימה מלאה לחלוטין מתקיים $n=\frac{3^{h+1}-1}{2}$, לכן $2n+1=3^{h+1}$ ונקבל כי גובה הערימה יהיה $h=log_3(2n+1)-1$
ד. ציינו מה עלות כמות העלים המינימלית ומה עלות העלים המקסימלית של ערימה טרינארית בגובה h(לא אסימפטוטית)
פתרון: כמות העלים המקסימלית היא אם הערימה מלאה, במצב כזה העץ יהיה עץ טרינארי מלא לחלוטין, מס' הקודקודים ברמה התחתונה יהיה 3^h.
כמות העלים המינימלית היא לא בהכרח 1. אם הערימה ברמה התחתונה עם עלה אחד, אזי שיש ברמה הקודמת עוד $3^{h-1}-1$ עלים. כיוון שהם גם נחשבים לעלים למרות שהם בברמה מעל. וכן אם נסתכל נראה שמס' העלים המינימלי תמיד יהיה $3^{h-1}-1+1=3^{h-1}$(באופן זהה, אם לוקחים 2 עלים ברמה התחתונה אז זה משאיר $3^{h-1}-2$ עלים למעלה).

שאלה 2:
הצע מבנה נתונים S שמתחזק n מפתחות שונים זה מזה ותומך ב:
חיפוש מפתח k $O(logn)$
הכנסת מפתח k בזמן $O(logn)$
מחיקת מפתח k בזמן $O(logn)$
$sum(S,k)$ החזרת סכום המפתחות במבנה S המקיימים שערכם קטן מk. זמן ריצה $O(logn)$
פתרון:
נשתמש כמובן בעץ AVL ונקבל הכנסה מחיקה וחיפוש בחינם.
הקאץ הוא הפעולה האחרונה, אך זה לא יהיה כזה קשה. נוסיף לכל איבר שדה שיתאר את סכום האיברים בתת העץ שלו (ניתן לפצל למשל לשני שדות - מימין ומשמאל, זה לא משנה), ראינו כבר בתרגול שהוספת שדות מסוג זה תעלה $O(1)$ ותשנה מס' קבוע של פעולות ולכן לא יהיה בעייתי אסימפטוטית לממשה. כעת לאחר שלכל איבר יש את השדה הנ"ל נסביר כיצד לממש את הפעולה האחרונה:
מה שנרצה יהיה לחפש את k. אם נמצא אותו מעולה, ייתכן שנגיע למצב והוא לא קיים בעץ. אם הוא קיים, אזי סכום המפתחות שערכם קטן מk זה תת העץ השמאלי שלו. (סכום תת העץ ששמרנו בתוך שדה מראש ועמלנו קשות.....), אחרת אנחנו נרצה לחפש את העוקב שלו בעץ. מדוע? אם 5 לא נמצא בעץ אבל 7 כן והוא האחד אחריו, כל מי שקטן מ7 יקיים את מה שהמפתח שלי רוצה.
לחפש עוקב בעץ זה $O(logn)$ -
אנחנו נלך לבדוק האם קיים לו תת עץ ימני, אם כן זה המינימום בתת העץ הימני. אחרת, נעשה למעלה בעץ עד שנגיע לקודקוד שאנחנו בן שמאלי שלו. סה"כ מצאנו את העוקב ובמקרה כזה נחזיר את ערך השדה של העוקב.
הבהרה - אנחנו מחפשים את העוקב ומחזירים את השדה שמתאים לתת העץ השמאלי שלו, כלומר מי שקטנים מהעוקב.
דרך נוספת (פחות כדאי) היא להוסיף את האיבר k לעץ, ואז לשלוף את סכום הקטנים ממנו, להחזיר, ובסוף למחוק אותו, אבל זה יותר עבודה.
סה"כ אנחנו מחפשים עוקב ואיבר ושולפים ערך שדה קבוע ולכן $O(logn)$. מש"ל.

שאלה 3:
א. מצא חסם הדוק אסימפטוטית של הפונקציה הבאה:

$$T(n) = \sqrt{2}T(\frac{n}{2}) + logn$$

הוכיחו קביעתם באינדוקציה.
פתרון: לאחר קצת פתיחה ומשחק באייפד מגיעים לנוסחה הבאה:

$$T(n) = (\sqrt{2})^i T(\frac{n}{2^i}) + \sum_{k=0}^{i-1} (\sqrt{2})^k log(\frac{n}{2^k})$$

כיף חיים. ננסה לסדר יפה את הביטוי מימין בסיגמה, נקבל

$$\sum_{k=0}^{i-1} (\sqrt{2})^k log(\frac{n}{2^k}) = log(n) + \sqrt{2}log(\frac{n}{2}) + \sqrt{2}^2 log(\frac{n}{4}) + ... + \sqrt{2}^{i-1} log(\frac{n}{2^{i-1}}) =$$

$$logn + \sqrt{2}logn - \sqrt{2}log2 + \sqrt{2}^2 logn - \sqrt{2}^2 log(4) + ... + \sqrt{2}^{i-1} log(n) - \sqrt{2}^{i-1} log(2^{i-1}) =$$

$$logn + \sqrt{2}logn + \sqrt{2}^2 logn + ... + \sqrt{2}^{i-1} log(n) - (\sqrt{2}log2 + \sqrt{2}^2 log(4) + ... + \sqrt{2}^{i-1} log(2^{i-1}))$$

$$= logn(1 + \sqrt{2} + \sqrt{2}^2 + ... + \sqrt{2}^{i-1}) - (\sqrt{2} + 2\sqrt{2}^2 + 3\sqrt{2}^3 + ... + (i-1)\sqrt{2}^{i-1})$$

הסכום מימין הוא הנדסי ונקבל

$$1 + \sqrt{2} + \sqrt{2}^2 + ... + \sqrt{2}^{i-1} = \frac{\sqrt{2}^i - 1}{\sqrt{2} - 1}$$

אנחנו רק מנסים לקבל אינטואיציה, ולכן

$$T(n) \leq (\sqrt{2})^i T(\frac{n}{2^i}) + logn \frac{\sqrt{2}^i - 1}{\sqrt{2} - 1}$$

אם היינו עושים $\frac{n}{2^i} = 1$ היינו מקבלים $i = logn$, ולכן כשנציב היינו מקבלים

$$T(n) \leq (\sqrt{2})^{logn} + logn \frac{\sqrt{2}^{logn} - 1}{\sqrt{2} - 1} = O(logn\sqrt{2}^{logn}) = O(logn(2^{logn})^{0.5}) = O(logn\sqrt{n})$$

בסיס: $T(1) = \sqrt{2} + log1 = \sqrt{2} = O(logn\sqrt{n})$
צעד: נניח כי קיים $c > 0$ עבורו לכל $n' < n$ מתקיים $T(n') \leq clog(n')\sqrt{n'}$. בפרט עבור $T(\frac{n}{2}) \leq clog(\frac{n}{2})\sqrt{\frac{n}{2}}$

$$T(n) = \sqrt{2}T(\frac{n}{2}) + logn \leq \sqrt{2}clog(\frac{n}{2})\sqrt{\frac{n}{2}} + logn = c\sqrt{n}log(\frac{n}{2}) + logn =$$

$$c\sqrt{n}logn - c\sqrt{n} + logn \leq c\sqrt{n}logn$$

אי השוויון נכון אמ"מ $-c\sqrt{n} + logn \leq 0$ כלומר $c\sqrt{n} \geq logn$, ובכן זה כמובן נכון החל מ $c = 1$. מש"ל.
ב. קבע יחס אסימפטוטי לבין n^{logn} לבין 2^{3logn}
פתרון: נשים לב כי $2^{3logn} = (2^{logn})^3 = n^3$, כעת נראה כי

$$lim_{n\to\infty}\frac{n^{logn}}{n^3} = lim_{n\to\infty}n^{logn-3}$$

עבור $n = 8$ מתקיים $logn = 3$ כלומר עבור $n = 8$ נקבל $n^{logn-3} = n^0 = 1$, אחרת, לכל $n > 8$ יתקיים $logn - 3 > 0$ כי log עולה ולכן נקבל שהגבול ישאף לאנסוף, סה"כ זה אומר כי $n^3 \in o(n^{logn})$.

שאלה 4:
א. תאר כיצד ניתן לממש מבנה שתומך בשתי מחסניות באמצעות מערך אחד כך שעלות כל פעולה לשיעורין היא $O(1)$, אם קיים מקום במערך ניתן להכניס לאיזה מחסנית שנרצה ללא הגדלת המערך.
ב. תאר כיצד ניתן לממש תור בשתי מחסניות כך שעלות כל פעולה לשיעורין היא $O(1)$
פתרון:
נתחיל מב' דווקא, קל יותר.
נשתמש בשתי מחסניות, אחת תהיה מחסנית הכנסות והשנייה הוצאות. נכניס את האיברים למחסנית S_1, בכל פעם שנרצה להוציא את האיבר שבראש התור נבצע n פעולות $s_2.push(s_1.pop)$, ואז נשלוף את האיבר המתאים החוצה. , סה"כ לשתי הפעולות שכאן נצטרך כל פעם 2 מטבעות, וכן נרצה להוציא גם את האיבר החוצה לכן נשלם סה"כ $3 = 2 + 1$ מטבעות לכל איבר בהכנסה, מה שיעלה $O(1)$ לשיעורין.
כעת לסעיף א',
נגדיר כי המערך בחציו השמאלי יוקדש למחסנית הראשונה ובחלקו הימני למחסנית השנייה. מה שנעשה כעת יהיה כמו בהרצאה - כאשר נרצה להכניס נכניס לאיבר הראשון ממנו מתחיל המערך שאליו אנחנו מתייחסים ונקדם את המצביע, כאשר נרצה למחוק נשים ערך -1 למשל או $-\infty$ או ∞ (זה לא משנה) שזה אומר שאין איבר במקום ונזיז את האינדקס אחד אחורה.
מה עושים מבחינה דינמית? כאשר נגמר המקום לגמרי במערך, נעשה מערך דינמי כמו שמכירים. כלומר, כפי שראינו בהרצאה להכפיל פי 2. אחרת, אנחנו נרצה לחלק את היחס בין המחסניות שונה, כך שנכניס איברים גם לתחום של המחסנית שעוד לא התרוקנה ונוריד לה את גודל המקום שתופסת.
דרך שקולה היא כאשר נגלה כי נגמר המקום באחת המחסניות מה שנעשה יהיה כמו במערך דינמי - להגדיל אותו פי 2, ראינו בהרצאה כי זה עולה O(1) לשיעורין. כאשר יש יותר מדי מקום - כלומר נגדיר איזשהו גודל שעבורו זה יותר מדי ואם יש יותר ממנו ריק, אז נחתוך את המערך פי 2 ונעתיק את האיברים בהתאם לחצאים השונים , סה"כ גם את זה ראינו שיעלה O(1) לשיעורין. סה"כ מערך דינמי + לחצות את המערך.

תל אביב 2023 **שאלה 1**
פתור נוסחת נסיגה הבאה

$$T(n) = 3T(\frac{n}{3}) + nlogn$$

פתרון:
בשביל האינטואיציה, אם נפתח נקבל

$$T(n) = 3^iT(\frac{n}{3^i}) + n\sum_{k=0}^{i-1}log(\frac{n}{3^k})$$

אם נפתח את הסכום נקבל

$$ilogn - (1 + 2 + 3 + ... + (i-1)) = ilogn - \frac{i(i-1)}{2}$$

$$T(n) = 3^i T(\frac{n}{3^i}) + inlogn - \frac{ni(i-1)}{2}$$

נעשה $\frac{n}{3^i} = 1$ ונקבל $i = logn$, נציב ונקבל

$$T(n) = n + nlog^2n - \frac{nlogn(logn-1)}{2} = O(nlog^2n)$$

כעת קיבלנו אינטואציה שכמובן איננה הוכחה. נוכיח באינדוקציה $T(n) = O(nlog^2n)$.
בסיס: עבור $n = 1$ נקבל $T(n) = 1 + 1log1 = O(nlog^2n)$ כי זה קבוע.
צעד: נניח נכונות לכל גודל קלט $n' < n$ ובפרט יתקיים שקיים $c > 0$ כך ש $T(\frac{n}{3}) \leq c\frac{n}{3}log^2(\frac{n}{3})$. כעת נקבל

$$T(n) = 3T(\frac{n}{3}) + nlogn \leq c\frac{n}{3}log^2(\frac{n}{3}) + nlogn = \frac{cn}{3}(logn-1)^2 + nlogn \leq$$

$$\frac{cn}{3}log^2n + nlogn \leq \frac{cn}{3}log^2n + nlog^2n = nlog^2n(\frac{c}{3}+1) \leq cnlog^2n$$

השוויון נכון אמ"מ $c \geq \frac{c}{3}+1$ כלומר $\frac{2c}{3} \geq 1$ כלומר לכל $c \geq 1.5$ לכן אם נקח $c = 2$ יתקיים הדרוש

דינמי

הבעיה: סדרת מספרים $w[1], ...w[k]$ תקרא מתחלפת אם לכל i זוגי $w[i] > w[i+1]$ ולכל i אי זוגי $w[i] < w[i+1]$. בעיית תת הסדרה המתחלפת -

קלט: סדרת מספרים A

פלט: אורך תת הסדרה המתחלפת הארוכה ביותר.

פתרון:

נגדיר $f(i, state)$ באשר עוברים על איברים $1, ...i$ כאשר בחרתי בi אחרון. וכן מצב קודם היה $state$. אם $state = 1$ עולים אחרת יורדים. נקבל

$$f(i, state) := \begin{Bmatrix} max_{1 \leq j < i}\{ & \begin{matrix} f(j,0)+1\} & A[i] > A[j] \wedge state = 1 \\ f(j,1)+1\} & A[j] > A[i] \wedge state = 0 \end{matrix} \end{Bmatrix}$$

אם $i = 0$ נגדיר באופן ריק תת סדרה באורך 1. כלומר $f(0,0) = f(0,1) = 0$. הפתרון יכול להיות בכל תא (מסתיימים בכל תא) לכן את המטריצה שניצור למלא שתעלה $n \times 2$ נסרוק למציאת המקסימלי שיהיה $max_{1 \leq i \leq n}\{A[i,0], A[i,1]\}$.
סה"כ מקום $O(n)$, בזמן $O(n^2)$ כי סורקים בכל תא $O(i)$ תאים.

שאלה 4 תל אביב

נתונים רשימת קטעים מהצורה $I = (a, b)$. כל קצותיו מס' טבעיים $0 \leq a_i \leq b_i \leq n^{10}$. תאר אלגוריתם שמחשב את האורך המכוסה ע"י הקטעים (קרי, מידת איחודם) ב$O(n)$.

למשל עבור $(2,5)(3,7)(9,10)$ יוחזר 6.

פתרון:

ראשית נתון גודל קלט, לכן נרצה למיין את האיברים לפי נקודת ההתחלה שלהם. נוכל לעשות זאת ע"י בסיס $R = n$ ואז $d = log_n n^{10} = 10$ ונקבל $O(10(n+n)) = O(20n) = O(n)$ למיונם.

כמו כן, לכל קטע נחשב את גודלו ע"י $L = b - a$.

כעת, נחזיק בקאונטר. כעת, נעבור על האיברים במערך הממויין. נסתכל על הראשון ונחשב את האורך שלו, נחלק למקרים:

אם האורך שלו לא יתנגש עם האיבר הבא - קרי, $a + L < a_2$ מעולה לנו, נוסיף לקאונטר את L.

אחרת, תהיה התנגשות. כלומר הקטע המקורי כעת יחולק לשניים $[a_1, a_2][a_2, a_2 + L_2]$. נוסיף לקאונטר את $a_2 - a_1$, ונמשיך הלאה עם a_2 לפי השלבים המתוארים.

כמו כן כעת נעיר שנצטרך להגדיר מחדש את נקודת ההתחלה, סה"כ במקרה כזה נרצה לבצע איחוד של הקטעים. לכן לקאונטר יתווסף $a_2 - a_1$, והקטע החדש יהיה $[a_2, max\{b_1, b_2\}]$ כך שנכיל את כולו.
סה"כ ריצה על n איברים שתעלה $O(n)$.

שאלה 4 2007 מהטכניון:
במערכת הבחירות בישראל יש n מפלגות המזוהות ע"י אינדקסים 0 עד $n-1$. בנה מבנה ש:
א. $init(V)$ מאתחל את המבנה. הפרמטר ל$init$ הוא מערך V בגודל n כך שבתא הi בו רשום מס' הקולות שקיבלה המפלגה הi בבחירות. V אינו ממויין. זמן נדרש: $O(n)$
ב. $AddVotes(i, votes)$ מוסיף עוד $votes$ קולות למפלגה i בזמן $O(logn)$
ג. $Threshold()$ מחיקת מפלגות שלא עוברות את אחוז החסימה (עומד כיום על 2%), כל מפלגה שלא קיבלה לפחות 2% מסך הקולות תימחק מהמבנה. סיבוכיות נדרשת $O(tlogn)$ כאשר t הוא מס' מפלגות שלא עוברות את אחוז החסימה מהמבנה.
ד. $coalition()$ בכדי להרכיב קואליציה צריך 60 מנדטים. אחד הדרכים להרכיב קואליציה היא ע"י צירוף המפלגות הגדולות זו אחר זו לפי סדר גודלן (הנמדד לפי מס' הקולות) עד שעוברים ממש את 60 המנדטים. עזרו לראש הממשלה להקים ממשלת אחדות. הפעולה תדפיס את האינדקסים של המפלגות החברות בקואליציה שכזו ממויינות לפי גודלן בסדר יורד. בסיום פעולה זו מבנה הנתונה ישאר כשהיה לפני הביצוע. זמן נדרש $O(clogn)$ כאשר c הוא מס' המפלגות החברות בקואליציה.
הבהרה: מס' המנדטים של מפלגה i בעלת x_i קולות שעברה את אחוז החסימה יחושב ע"י הנוסחה $120 \times \frac{x_i}{\sum x_i}$ בערך שלם תחתון.
מותר לכם להשתמש בכל מה שנלמד במסגרת הקורס.
הנחות: מותר לקרוא ל$coalition$ רק לאחר קריאה ל$threshold$. הניחו כי אין שתי מפלגות עם מס' מנדטים זהה.

פתרון:
איזו שאלה נחמדה. סיפרו לנו סיפור יפה ואם נסנן אותו אין כאן יותר מדי.
נבנה ערימת מקסימום וטבלת האש. הרעיון יהיה לנסות להתגבר על הבעיה בערימה, שהיא שאסור לחפש ולמחוק. הרעיון יהיה פשוט ואז אסביר אותו בהתאם למבנה:
יודעים לבנות ערימה ב$O(n)$, כמו כן נבנה גם טבלת האש ב$O(n)$. כל איבר יכיל פוינטרים דו כיוונים. כאשר נרצה לחפש איבר נחפשו בטבלת האש ב$O(1)$, ונשתגר עם הפוינטר לערימה. כאשר נרצה למחוק נחפשו, ואז נחליפו עם האיבר שנמצא ברמה התחתונה מימין, נמחק אותו ונעשה $heapify$.
כעת למבנה –
נרצה לבנות ערימת מקסימום לפי מס' הקולות, עם פוינטרים דו כיוונים המכוונים לתוך טבלת האש. בנייה תעלה $O(n)$. כמו כן יהיה מצביע מקושר למערך שקיבלנו לתא הi המתאים.
הוספת קולות – נלך אל המערך, לתא הi, משם נשתגר לטבלה ולערימה. כפי שהסברנו קודם נמחק את הערך שהתאים קודם, נסמנו x ונכניס את הערך $x + votes$. הכנסה לערימה עולה $O(logn)$ וגם מחיקה בשיטה שהסברתי כאן, וכן לטבלת האש $O(1)$. סה"כ $O(logn)$
מחיקת מפלגות – כעת נחליט שבעת האתחול יהיה קאונטר ובו מס' הקולות הנוכחי, כאשר נעשה פעולת הוספת קולות יתווספו גם אליו לסה"כ הקולות, כעת נחשב ממנו 2% ונחפש בערימה את המפלגות שסה"כ הקולות שלהן קטן מערך זה. כמו כן נחליט כרגע שנחזיק במקביל גם ערימת מינימום על מנת לדעת איזה הכי קטנים להוציא (אם בכלל), זה יהיה קל לתחזוק במקביל. סה"כ בעת הוספת קולות נגיע לאחת הרלוונטית עם פוינטר, ובעת הרכבת הקואליציה גם כן נחפש את המפלגה שנרצה ונוציא אותה (מחפשים עם האש).
קואליציה – הרבה קולות=הרבה מנדטים, ולכן נרצה ראשית לצרף לקואליציה את אלו עם מס' הערכים הכי גבוה. נבצע c הוצאות של המקסימום, נחשב עבורם את המנדטים ונוסיף לקאונטר. שנעבור אותו – נפסיק להוציא מפלגות. סה"כ $O(clogn)$.

טכניון 2007 שאלה 2
מערך נקרא ממויין בכאילו אם קיימים לך לכל היותר $logn$ זוגות מפרים. זוג מפר הוא זוג מהצורה $(i, i+1)$ בו $i+1 > i$ אך $A[i] > A[i+1]$. כלומר יש לכל היותר $logn$ אינדקסים כך ש$A[i] > A[i+1]$. הצע אלגוריתם שמקבל מערך ממויין בכאילו וממיין אותו ב$O(nloglogn)$
פתרון:
ראשית נמצא את הזוגות המפרים ונוציא את האינדקסים שלהם החוצה. קיבלנו מערך ממויין בלעדיהם. כעת נרצה למיין את המערך עם $logn$ הזוגות. נראה כי בהינתן $logn$ זוגות יש לכל היותר $2logn$ איברים שעלינו למיין. מיון זה יעלה $O(logn * loglogn)$, סה"כ קיבלנו שני מערכים ממויינים, נאחדם יחד לממויין ב$O(n)$ וסה"כ קיבלנו $n + lognloglogn \leq n + nloglogn = O(nloglogn)$.
הסבר מפורט יותר – עוברים לינארית על המערך ומוציאים את הזוגות הבעייתיים המקיימים $A[i] > A[i+1]$. זה יעלה לינארית. נסמן בהם אנסוף נניח ולא נרצה להתחשב בהם עוד כי הם במערך אחר. כעת נעתיק למערך אחר גם את אלו שהיו ממויינים – יש שם מערך ממויין כרגע, סה"כ לאחר המיון שהסברתי נקבל שני מערכים ממויינים

ונאחדם לאחד ב$O(n)$ ע"י השוואת הראש כל פעם והתקדמות האינדקס.

שאלה 2 אלגוריתמים 2021 סמס א

במדינת אלגו לנד החליטו להעניק השכלה כללית לכל התושבים. לצורך כך יצרו תואר שמכיל n קורסי מבואות ממגוון נושאים עם נקודות הזכות $z_1, ..., z_n$. כמות נקודות הזכות של כל קורס היא לכל היותר 6. כמות נקודות הזכות הנדרשות עבור כל הקורסים היא $L = \sum_{i=1}^{n} z_i$. כמות הסמסטרים לסיים את התואר היא s. עזרו לאזרחי אלגו לנד לחלק את הקורסים לסמסטרים כך שבכל סמסטר יבצעו קורסים שסך נקודות הזכות שלהם בסמסטר הוא לכל היותר $\frac{L}{s} + 6$. אין תלות בין הקורסים.

קלט: מערך n קורסים עם נקודות הזות ומס s של סמסטרים

פלט: חלוקה של האינדקסים לs קבוצות כך שלכל קבוצה יתקיים שסכום נקודות הזכות קטן מהמתואר מעלה.

תאר אלגוריתם חמדני שפותר את הבעיה והוכח נכונותו.

פתרון:

נמיין את הקורסים בסדר כלשהו, למשל עולה. נעבור על כל קורס ונכניס אותו לסמסטר הראשון שיש בו מקום.

הוכחה: נוכיח שזה יתן לנו פתרון חוקי ואופטימלי.

1. אכן גודל כל קבוצה יהיה בגודל הנדרש - בדקנו זאת באלגוריתם בכל שלב.
2. נב"ש שקיים קורס k שאין לו מקום. אזי, כל הקורסים מלאים עד $\frac{L}{s} + 6 - k + 1$. כלומר סה"כ מס' נקודות הזכות יהיה

$$s(\frac{L}{s} + 6 - k + 1) = L + s(7 - k)$$

נשים לב כי $k < 7$ ולכן $-k > -7$ ולכן $7 - k > 0$ וכן $s > 0$ וקיבלנו $L + s(7 - k) > L + 0 * s = L$ בסתירה, כי מס' נקודות הזכות הוא בדיוק L.

שאלה 4 קיץ 2017 אלגו מועד א

נסמן $A = \{a_1, ..., a_n\}, B = \{b_1, ..., .b_n\}$ שתי קבוצות מספרים שונות. כולם חיוביים ללא אפס. לכל $1 \le i < j \le n$ מתקיים $a_i \neq a_j, b_i \neq b_j$. הצע אלגוריתם חמדני אשר מוצא פרמוטציה (פונקציה חח"ע ועל) כך שהסכום $\sum_{i=1}^{n} \frac{a_i}{b_{f(i)}}$ הוא הקטן ביותר.

פתרון:

נראה כי כיוון שכל האיברים שונים בתוך הקבוצות ובין הקבוצות, נוכל להגדיר פונקציה הפיכה שכזו. כיצד נמזער את הסכום? אם נגרום ל$b_{f(i)}$ להיות הכי קטן. כיצד נעשה זאת? נמיין את A ואת B. כעת הם בסדר עולה. נסתכל על האיבר האחרון ביותר בB, אם נגרום לפונקציה לקבל ערך כלשהו קטן מאוד, המינימלי של A, ולהחזירו כפלט זה יהיה טוב עבורנו.

כלומר, פורמלית. נסמן את המינימום בA בm, ואת המקסימום בB בM. בהכרח כעת $A[0] = m, B[n] = M$. אם נגדיר את הפונקציה עבורם להיות $f(m) = M$, אזי מה שיקרה הוא שהפונקציה תקבל כקלט את המספר הכי קטן, וזה יהיה גם המינימום בa_i, והיא תחזיר לו קלט ממש גדול, בפרט $f(i)$ יהיה ממש גדול, ולכן האינדקס $b_{f(i)}$ יהיה מאוד גדול, ובמערך הממויין החדש גדול יותר זה אומר שגם בערכו הוא הכי גדול, ואז חילקנו הכי קטן בהכי גדול. וכך כמובן נתקדם פנימה לתוך המערך.

סה"כ סיבוכיות זמן הריצה תהיה $O(nlogn)$ למיונם, וn להרכיב את הפונקציה ולכן סה"כ $O(nlogn)$.

ליט קוד 152:

בהינתן מערך מספרים A כאשר $\forall a_i \in A : a_i \in \mathbb{R}$. מצא את תת המערך הרציף שמכפלת איברים מקסימלית

פתרון:

נגדיר $maxDp$= מכפלה מקסימלית של תת מערך שמסתיים באינדקס i. $minDp$- מכפלה מינימית של תת מערך שמסתיים בערך i.

בסיס יהיה $maxDp[0] = minDp[0] = A[0]$

נוסחאות הנסיגה:

$$maxDp[i] = max\{A[i], maxDp[i-1] * A[i], minDp[i-1] * A[i]$$

$$minDp[i] = min\{A[i], maxDp[i-1] * A[i], minDp[i-1] * A[i]$$

מדוע זה הפתרון? נראה כי יש שלוש אפשרויות בכל שלב: להתחיל תת מערך חדש באינדקס נוכחי, להמשיך מקסימום קודם, להמשיך מינימום קודם (חשוב אם המספר שלילי). נבנה מערך ב$O(n)$ זמן ומקום, נוכל להקטין מקום.

2023 קיץ ב':

שאלה 1:

נתון מערך $A[1,....n]$ עם n מספרים טבעיים שונים זה מזה המקיים את התנאי הבא: לכל $1 \leq i < j \leq n$ מתקיים אם $A[i] > A[j]$ אזי $j = i + 1$.

א. איזה אלגוריתם מיון ימיין את A בסיבוכיות זמן טובה יותר? מיון הכנסה או מיון מיזוג?

ב. האם התשובה בסעיף א' תשתנה אם הערכים בA אינם בהכרח שונים זה מזה?

פתרון:

א. אנו יודעים מהנתון שאם המערך יורד, זה רק בערכים צמודים, כלומר $A[i] > A[i+1]$. אחרת, יתכן שהמערך יעלה. במצב כזה, עדיף לנו להשתמש במיון הכנסה. מדוע? המקרה הגרוע ביותר לכל שני איברים במקומות סמוכים מתקיים $A[i] > A[i+1]$. ככל שנתקדם הערכים בזוגות הבאים יהיו גדולים יותר מהערכים בקודמים, לכן במיון הכנסה תתבצע החלפה אחת בכל בדיקה שניה ויהיו $\frac{n}{2}$ החלפות כלומר סדר גודל של $O(n)$ למיון, בעוד שמיון מיזוג תמיד יבצע מיון של ימין של שמאל ובאמצע ויאחד מה שתמיד יביא בהפרד ומשול ל$O(nlogn)$

ב. אם הערכים יכולים להיות זהים, למשל מערך שלכל $i \in A$ מתקיים $A[i] = a$, זה לכאורה המקרה הגרוע, עדיין יתבצעו בסה"כ n השוואות כי בכל פעם נבדוק איבר חדש עם הראשון משמאלו ושוב מיזוג תמיד יביא $O(nlogn)$.

שאלה 2:

א. נתון תור קדימיות Q שבאמצעותו ניתן לבצע שתי פעולות - הכנסת איבר ומחיקת מינימום. הוכח הפרך - לכל $m \in \mathbb{N}$ קיים $n > m$ כך שעבור n חייבת להיות לפחות סדרה אחת של n פעולות הכנסה והוצאת מינימום כך שלפחות אחת הפעולות מתבצעת בזמן $\Omega(logn)$

הוכחה:

נב"ש שקיים $n \in \mathbb{N}$ כך שכל אחת מהפעולות מתבצעת בזמן $o(logn)$. נשתמש בQ ונבצע n הכנסות למבנה הנתונים, ולאחר מכן נבצע n הוצאת מינימום. סה"כ עלה לנו $2n * o(logn) = o(nlogn)$, קיבלנו מיון מבוסס השוואות ב$o(nlogn)$ בסתירה לחסם תחתון.

ב. מה מס' ההשוואות שמבצעת הפונקציה שמקבלת מערך ומחזירה את המערך כערימת מקסימום, על מערך ממויין (מהקטן לגדול)? הניחו שקיים k טבעי כך ש $n = 2^k - 1$. בטאו תשובתכם באמצעות k.

פתרון:

מהנתון, נראה כי זה אומר שגובה העץ הוא $k-1$ והוא עץ שלם. כלומר, כל הרמות מלאות. גובה הערימה הוא $h = k - 1$. כעת, הפונקציה מתחילה מלמטה ומבצעת $heaipfy$ על המערך, המערך ממויין מהגדול לקטן, ולכן בבכל שלב הוא יצטרך לקדם את המקסימום מעלה. סה"כ כגודל מס' האיברים יתבצעו החלפות.

כמה עלים בעץ? נראה כי אם גובה העץ הוא $k - 1$, אזי יש ברמה התחתונה 2^{k-1} עלים. כל אחד מהם יצטרך להיות מוחלף לאורך הדרך , הדרך היא $k - 1$ באשר יגיע למעלה ולכן מס' ההשוואות יהיה $(k-1)2^{k-1}$.

שאלה 3:

נתונים שני עצי B עם פרמטר m. העץ T_1, T_2 בהתאמה בגובה h וכן כל הערכים בT_1 קטנים מהערכים בT_2. הצע אלגוריתם טוב ביותר מבחינת זמן ריצה שיאחדם.

פתרון:

הרעיון יהיה לאחד אותם בזמן $O(logn)$! אנחנו נמצא ערך k שיקיים שהוא גדול מכל מי שבT_1 וקטן מכל מי שבT_2. נרצה לשים את T_1 משמאל ואת T_2 מימין ולכן נמצא את המקסימום בT_1 והוא יהיה האבא המשותף שלהם או לחלופין המינימום בT_2 (גם אותו ניתן למצוא כי זה עץ חיפוש והוא הכי שמאלי בעץ), נקבל ערך אחד כשורש ושני תתי עצים שכל אחד מהםהבגובה h.

אם הערך שהוצאנו המקסימום/מינימום לא הקטין את גובה העץ זה יעבוד ללא בעיות, יהיה שני תתי עצים בגובה h וסה"כ עץ B תקין

אם כן הקטין את גובה העץ, נבצע עוד הכנסה זמנית של איבר, על מנת שנוכל למצוא אחד בניהם, נסדרם ולאחר מכן שנקבל עץ B נוכל לחפש את הערך הלא טוב ולמחוק אותו (מותר אסימפטוטית הרי לא ישנה דבר)

סה"כ $O(logn)$.

שאלה 4:

א. הוכח הפרך: המפתח k נמצא במבנה נתונים מילוני שמיושם באמצעות טבלת האש כלשהי.

האם הוצאת המפתח k ממבנה הנתונים ולאחר מכן הכנסנו תשנה את מבנה הנתונים או שלא?

פתרון: הפרכה. נעבוד עם גיבוב קוקייה. ראינו כבר בהרצאה דוגמאות רבות לשינויים מסוג זה.

ב. נכון או לא נכון: לכל f, g מהטבעיים לטבעיים אם $log(f(n)) = O(log(g(n))$ אזי $f(n) = O(g(n))$
הפרכה: נקח $f = n^2$ ו-$g = n$.

אלגו 1 חמדניים 2014 תרגיל 6 שאלה 4

למסעדת החלומות צפויים להגיע n סועדים. הסועד ה-i מגיע בזמן r_i ונדרש זמן p_i של השף בכדי להכין לו את המנה שהזמין. השף אינו מתחיל בהכנת המנה של הסועד לפני הגעתו למסעדה. בנוסף לשף שמורה זכות להחליט אם להקפיא את הטיפול במנה של סועד מסויים לטובת סועד אחר ולהמשיך בהכנתה במועד מאוחר יותר.
יהי w_i משך ההמתנה של הסועד ה-i להכנת מנתו. מטרתנו היא לתכנן לוח זמנים של השף כך שזמן ההמתנה הממוצע יהיה מינימלי כלומר להקטין ככל הניתן את $\frac{\sum_{i=1}^{n} w_i}{n}$. כל הפרמטרים עבור n הסועדים ידועים מראש.
כתוב אלגוריתם חמדני לתכנון לוח הזמנים של השף, והוכח נכונותו.

פתרון:

הרעיון יהיה להקטין ככל הניתן את סכום ההמתנות. השף יודע את זמני ההגעה של הלקוחות ויודע את הזמן שיקח לו לבצע את המנה. ולכן נרצה למיין את הנתונים בדרך כלשהי. המטרה תהיה למיין את המנות בהתחשב בזמני ההגעה וזמני ההכנה. בכל עת השף יכין את המנה שזמן הכנתה הקטן ביותר מבין ההזמנות שהוא יכול לקחת כעת. כלומר, למיין לפי p_i ובכל שלב לבצע את המנה עם זמן הביצוע הקטן ביותר מבין המנות של הלקוחות שהגיעו.
סה"כ מעבר לינארי $O(n)$ ועוד $O(nlogn)$ למיון נקבל $O(nlogn)$.
כיצד נממש? נשתמש בערימת מינימום עבור הלקוחות שהגיעו, כלומר נבנה ערימת מינימום, בכל שלב נכניס את האנשים שהגיעו ונשלוף את זה שזמן ההכנה שלו הקצר ביותר. זה לא יפגע בסיבוכיות.
כעת מי אמר שזה בכלל מביא פתרון אופטימלי?
טענה 1. למת הבחירה החמדנית - קיים פתרון אופטימלי בו בכל שלב נקח את המנה עם זמן ההכנה הקצר ביותר.
נמיין את המנות ונסמנן $x_1, ..., x_n$ לפי זמני ההכנה. כעת נסתכל על המנה הראשונה x_k שזמנה הקצר ביותר מבין ההזמנות שכעת ניתן לטפל בהם. יהי פתרון אופטימלי OPT. אם $x_k \in OPT$ סיימנו. אחרת, x_k היא לא המנה הראשונה שכרגע מטפלים בה, כלומר קיימת מנה אחרת מהמנות שכרגע הגיעו, x_w כך ש-$p_w > p_k$. נסתכל על הפתרון הבא:

$$OPT' = OPT/\{x_w\} \cup \{x_k\}$$

כלומר כמובן שגם המנה x_w תטופל בהמשך הכוונה היא שנסתכל על הפתרון בו x_k היא המנה הראשונה. אזי, ראשית כמובן שזה פתרון חוקי.
שנית, נרצה להוכיח שמדובר בפתרון אופטימלי.
ובכן, נראה כי כתוצאה מהזזת x_w, זמן ההמתנה של x_w יגדל ב-p_k וזמן ההמתנה של x_k יקטן ב-p_w, מדוע? כי החלפנו בין הזמנים. כעת נראה כי

$$|OPT|' = |OPT| - p_k - p_w + (p_w + p_k) + (p_k - p_w) = |OPT| + p_k - p_w < |OPT|$$

מדוע? גודל הפתרון האופטימלי פחות שני זמני החלוקה, ועוד זמני החלוקה החדשים. וכן $p_k < p_w$ ולכן $p_k - p_w < 0$.
סה"כ נקבל פתרון אופטימלי.
טענה 2. למת תת המבנה האופטימלי - פתרון בו בכל שלב נקיים את 1. (נבחר את המנה עם זמן ההכנה הקצר ביותר) יחד עם פתרון לכל שאר המנות אחריה, הוא פתרון אופטימלי לבעיה.
הוכחה: יהי פתרון אופטימלי A. נב"ש שקיים פתרון אופטימלי יותר B כלומר $|B| > |A|$. בהכרח, מלמה 1, x_k יופיע בפתרון האופטימלי קודם. כלומר $x_k \in A, B$. בהכרח, לפי ההנחה, $|A/\{x_k\}|$ (שוב, להזיז את זמנו) הוא פתרון אופטימלי. לכן

$$|A/\{x_k\}| < |B/\{x_k\} \Longrightarrow |A| - |\{x_k\}| < |B| - |\{x_k\}| \Longrightarrow |A| < |B|$$

בסתירה. נבהיר שהכוונה בהורדת x_k מהפתרון זה להזזת הזמן, שהראינו שמקצרת.

אלגו 1 2018 תרגיל 6 חמדן

תהיק בוצה S של n מקטעים על הישר הממשי. הקבוצה S מיוצגת ע"י זוג מערכים R, L באורך n כאשר מקטע i בקבוצה מיוצג עי $R[i], L[i]$. שמייצגים בהתאמה ערך נקודת קצת שמאלית וערך נקודת קצה ימנית. צביקה תקינה

של S היא צביעה בה ניתן לכל אחד מהמקטעים בS צבע כך שאין זוג מקטעים חופף בעל אותו צבע. כתוב אלגוריתם חמדני אשר מוצא את המס' המינימלי של צבעים הדרוש על מנת ליצור צביעה תקינה והוכח נכונותו.

פתרון:

בעצם הרעיון הוא למצוא את מס' החפיפות. האידיאל מבחינתנו יהיה לצבוע את כל המקטעים באותו צבע וסיימנו אך ייתכנו חפיפות.

נרצה למיין את נקודות ההתחלה של המקטעים. כעת, בהינתן שאני בנקודת התחלה נבדוק האם נקודת הסוף שלי מסתיימת לפני תחילת המקטע הבא אחרי, אם כן נצבע בצבע הבסיסי לבן למשל שנחליט שבו נרצה לצבוע את כולם. אחרת, נצבע אם יש חפיפה את שניהם בצבע אחר, x_1, x_2 נוסיף לקאונטר ונתקדם. כך באופן זהה אם נתקל בשלוש חפיפות ויותר. סה"כ מעבר לינארי ומיון נקבל $O(nlogn)$

שאלה 1:

תהא A ערימת מקסימום בגודל n. יהי z צומת בערימה בעומק k. (הערה - עומק השורש הינו אפס, עומק קודקוד אחר הוא +1 מעומקו של ההורה). מוסיפים לכל אחד מאיברי תת הערמה של A המושרשרת בz את הקבוע $c > 0$. ברצוננו לתקן את המבנה כך שיחזור להיות ערימת מקסימום (ללא שינוי יתר הערכים במבנה A, אלא רק בתת העץ המושרש בz)

א. כתוב אלגוריתם לתיקון המבנה A שזמן ריצתו $O(klogn)$, הסבירו נכונות (אין צורך להוכיח)

ב. כתבו אלגוריתם לתיקון המבנה A שזמן ריצתו הוא $O(2^{logn-k}logn)$. הסבירו את נכונות האלגוריתם.

פתרון:

א. נראה כי התכונה היחידה של הערימה שנפגע היא שהאבא גדול מבניו, לכן נסתכל על הקודקוד שבראש הערימה z ונפעפעו מעלה עד שיגיע למיקום הנכון. סה"כ הפעפוע עלה $O(logn)$. כמה פעפועים כאלו נדרש לעשות? k כ פעפועים, במקרה הגרוע ביותר נצטרך לבצע k פעפועים כאלו כגובה הערימה.

ב. ננסה להבין מה הסיבוכיות אומרת, $logn$ זה גובה העץ, k זה עומק הערימה,ולכן $logn - k$ זה גובה תת הערימה z. 2^{logn-k}זה מס' האיברים בתת הערימה הזו. סה"כ לכל אחד מהאיברים בתת הערימה נבצע $heapify$ שיעלה $O(logn)$ ונקבל ערימה תקינה.

תכנון דינמי:

נתונות K סדרות, $s_1, ...s_k$. גודל כל אחת מהן הוא n. נרצה למצוא את תת הסדרה המשותפת הארוכה ביותר המקסימלית שלהן.

פתרון:

נגדיר $f(i_1, ..., i_k)$ כאורך תת המחרוזת המשותפת הארוכה ביותר של הסדרות כאשר לכל סדרה j מסתכלים על האיברים $1, ...i_j$. כעת נראה כי הנוסחה הרקורסיבית תהיה

$$f(i_1, ..., i_k) = \left\{ \begin{array}{cc} 0 & i_1 \vee .... \vee i_k = 0 \\ 1 + f(i_1 - 1, ..., i_k - 1) & S[i_1] = ... = S[i_k] \\ max\{f(i_1 - 1, i_2, ..., i_k), f(i_1, i_2 - 1, ..., i_k), ..., f(i_1, i_2, ..., i_k - 1)\} & o.w \end{array} \right\}$$

פתרון תכנון דינמי: נאלץ ליצור מערך k-מימדי, כל אחד מממדיו יהיו n. סה"כ נקבל $O(n^k)$ לזמן ולמקום.