פתרון תרגיל בית יבש מס׳ 3

1

בכל רגע הקטע נראה כך: 

לכן מספיק לשמור את סדרת התחלות הקטעים (הנקודות השמאליות ביותר). אז כל קטע המתחיל בנקודה אי-זוגית הינו לבן והשאר שחורים. תחת ההנחה שכל הקטעים הם חצי פתוחים (כלומר, הקטע הכולל הינו $[0,1)$ וגם פעולת Reverse(x,y) מתבצעת על קטעים מהצורה $[x,y)$), צבע של נקודה הנמצאת בין שתי נקודות קצה נקבע לפי צבע של נקודה הקצה השמאלית.

נממש את המבנה כ-Rank Tree על בסיס עץ AVL (בדיוק כמו מוגדר בתירגול). בצמתים נשמור את הקואורדינטות של נקודות הקצה.

- Init
  נבנה Rank Tree המכיל צומת בודד בעל ערך 0 (נקודת הקצה של הקטע הגלובלי).

- Reverse(x,y)
  נחפש במבנה את הנקודה x.

  - אם הנקודה לא נמצאת במבנה, אזי נכניסה למקום המתאים ע״י הכנסה רגילה ל-Rank Tree.
  - אם הנקודה כן נמצאת במבנה, הרי תת קטע שמתחיל ממנה נצבע בצבע של תת הקטע הקודם לו, לכן יש לאחד את שני הקטעים. לשם כך נמחק את הנקודה x מהמבנה.

  בזה נהפוך אוטומטית את הצבעים של כל הנקודות החל מ-x ועד הסוף, כי עקב הפעולה תשתנה זוגיות של כל תת קטעים החל מ-x.
  כעת נבצע פעולה סימטרית עבור הנקודה y.
  סה״כ יש לבצע שתי פעולות רגילות של הכנסה/הוצאה ל-Rank Tree, כלן סיבוכיות הפעולה Reverse(x,y) היא $O(\log n)$.

- Color(x)
  נבצע חיפוש אחר נקודה x במבנה.

  - אם נמצאה, נחשב את ה-rank שלה.
  - אחרת, נסמן ב-y את העלה בו עצרנו בחיפוש. אזי y הינו הערך הקרוב ביותר ל-x משמאל או מימין. לכן נמצא את ה-rank של y. ואז ה-rank של x שווה ל-rank של y, אם $x > y$. אחרת, ה-rank של x שווה ל-rank של y מינוס 1.

  נקבע את צבעה של הנקודה לפי זוגיות של ה-rank שלה (אי-זוגי – לבן, זוגי – שחור).
  סיבוכיות הפעולה היא $O(\log n)$.

2

נממש את המבנה כ-Union-Find על בסיס עצים הפוכים. בנוסף, נשמור אברי כל קבוצה ברשימה מקושרת מעגלית (או דו-כיוונית).
נחזיק מערך M בגודל $n \times n$ אשר בכל תא שלו נשמור רשומה בעלת השדות הבאים: col עבור צבע של המשבצת המתאימה, fat עבור מצביע (אינדקס במטריצה) לאבא בעץ הפוך, num עבור מספר האיברים

בקבוצה (רלוונטי רק לאיברים שהם שורשי הקבוצות שלהם) ו-$next$ עבור מצביע לאיבר הבא ברשימה.
בנוסף, נגדיר משתנה NUM שיכיל את המספר הכולל של הקבוצות במבנה.

- Init – נאתחל את המשתנה NUM: $NUM \leftarrow 0$.
  נאתחל את המטריצה M, כלומר: בכל תא שלה $[i,j]$ נבצע:
  - $col \leftarrow \text{White}$ כל המשבצות לבנות בהתחלה.
  - $fat \leftarrow (-1,-1)$ כל משבצת הינה קבוצה בפני עצמה, לכן אין לה אבא.
  - $num \leftarrow 1$ כל קבוצה מכילה משבצת בודדת.
  - $next \leftarrow (i,j)$ כל רשימה מעגלית מהווה לולאה עצמית.

  את האיתחול ניתן לבצע ב-$O(n^2)$ באופן טריוויאלי.
  כדי לבצעו ב-$O(1)$ יש להשתמש בשיטה שנלמדה בכיתה לאיתחול מערך ב-$O(1)$: בנוסף למטריצה המקורית נשמור עוד אחת באותו גודל ועוד מערך חד-מימדי באורך n^2 (מחסנית). במחסנית נשמור את האינדקסים של התאים בהם כבר נעשה שימוש במטריצה M ובמטריצה הנוספת נשמור מצביעים לערכים המתאימים בתוך המחסנית, כדי לאפשר את כל הבדיקות ב-$O(1)$.

- Color(x,y) –

  א. ניגש ל-$M[x,y]$ ונבדוק את צבע המשבצת.
  אם המשבצת כבר שחורה, נסיים.
  אחרת, נצבע אותה בשחור ע"י $col \leftarrow \text{Black}$ ונעדכן את מספר הצורות $NUM \leftarrow NUM+1$.

  ב. עבור כל אחת משמונה השכנות של $M[x,y]$ נבדוק את הצבע.
  אם המשבצת (נסמנה $M[i,j]$) לבנה, נעבור לשכנה הבאה.
  אחרת, נבצע $Union(Find(x,y), Find(i,j))$, נאחד את הרשימות המעגליות של שתי הקבוצות ונקטין את מונה הקבוצות: $NUM \leftarrow NUM-1$.

  סיבוכיות של פעולת $Find$ הינה $O(\log n)$ במקרה הגרוע (במימוש המשופר) ופעולת $Union$ לוקחת $O(1)$ זמן. לכן הסבוכיות הכוללת של Color(x,y) היא $8(2\cdot O(\log n^2)+O(1)) = O(\log n)$.

- Shape(x_1,y_1,x_2,y_2) – נחזיר "כן", אם $Find(x_1,y_1) = Find(x_2,y_2)$. "לא" – אחרת.
  הסבוכיות במקרה הגרוע היא $2\cdot O(\log n^2) = O(\log n)$.

- Which(x,y) – ניגש למשבצת $M[x,y]$, אם היא לבנה, נסיים (המשבצת לא שייכת לאף צורה).
  אחרת, נעבור על הרשימה המעגלית החל מ-$M[x,y]$ ועד שנתקל בה שנית. תוך כדי מעבר נדפיס את האינדקסים של המשבצות.

  סיבוכיות הפעולה הינה $O(k)$ כאשר k הוא אורך סדרת הפלט. במקרה הגרוע זה $O(n^2)$.

- Num – נדפיס את ערכו של NUM. סיבוכיות $O(1)$.

הסיבוכיות הממוצעת של איתחול, C פעולות Color, S פעולות Shape ו-N פעולות Num הינה:

$$O(1) + O(C\cdot\alpha(C,n)) + O(S\cdot\alpha(S,n)) + N\cdot O(1) = O((C+S)\alpha(\ldots)+N) \approx O(C+S+N)$$

הערות :

א. שימו לב כי C, S ו-N <u>אינם</u> קבועים.

ב. מימוש של Union-Find בעזרת רשימות מקושרות לא משיג את הסיבוכיות המינימלית: אמנם סיבוכיות של Shape במימוש זה היא טובה יותר ($O(1)$), אבל סיבוכיות של Color אז תהיה הרבה יותר גרועה ($O(n^2)$ במקרה הגרוע). וגם הסיבוכיות הממוצעת במימוש זה לא תגיע לזמן כמעט קבוע לכל פעולה.

3

נממש את המבנה באמצעות שתי ערימות:

- ערימת max שתכיל את $\left\lceil \frac{n}{2} \right\rceil$ האיברים הקטנים יותר (כלומר, קטנים או שווים לחציון).
נסמן את הערימה ב-S.

- ערימת min עבור $\left\lfloor \frac{n}{2} \right\rfloor$ האיברים הגדולים יותר (גדולים מהחציון).
נסמן את הערימה ב-L.

בנוסף לשדות הרגילים של הצמתים בערימות, יהיה בכל צומת של הערימה S שדה min שיכיל את הערך המינימלי בתת-העץ של הצומת (כולל עצמו) ובכל צומת של הערימה L שדה max לשמירת הערך המקסימלי בתת-העץ של הצומת. נחזיק גם משתנה בולאני $balance$ אשר יקבל ערך 0 כאשר שתי הער-ימות מכילות אותו מספר של איברים, וערך 1 אם ב-S יש יותר איברים (ב-1) מאשר ב-L.

את הפעולות יש לממש באופן הבא:

- init –

א. מצא את החציון של האיברים הנתונים בעזרת הפרוצדורה Select(A,first,last,i) דטרמיני-יסטית שנלמדה בכיתה. הפרוצדורה מאפשרת מציאת חציון (בעצם, ערך i-י בגודלו) מתוך n מספרים בסיבוכיות זמן $O(n)$.

ב. ע"י השוואה עם ערך החציון הפרד את כל האיברים לשתי רשימות: אחת – של מספרים קטנים או שווים לחציון, ושניה תכיל את כל אלה הגדולים מהחציון. סיבוכיות ההפרדה היא $O(n)$.

ג. כעת בצע make_heap על כל אחת מהרשימות. הסיבוכיות היא פעמיים $O(\frac{n}{2})$.

ד. בצע סיור postorder בערימה S ותוך כדי סיור עדכן את השדה min בכל צומת v באופן הבא: אם v הוא עלה, אזי $min(v) \leftarrow val(v)$ (כל עלה הוא המינימום בתת העץ שלו); אחרת, $min(v) \leftarrow \min\{min(left\_son(v)), min(right\_son(v))\}$.
בצע פעולה סימטרית לעדכון השדות max בצמתי הערימה L.

ה. אתחל את $balance$ ל-0 אם לשתי הערימות יש מספר זהה של האיברים, אחרת $balance \leftarrow 1$.

הסיבוכיות הכוללת הינה $O(n)$.

- insert(x) – השווה את x עם החציון (הנמצא בשורש של S).
בצע $balance \leftarrow 1 - balance$ (עקב הכנסת איבר חדש האיזון בין הערימות חייב להשתנות).

  - אם x קטן או שווה לחציון, הכנס את x לערימה S.
אם $balance = 0$ הוצא מ-S את שורשה והכנסו ל-L.
  - אחרת, הכנס את x לערימה L.
אם $balance = 1$ הוצא מ-L את שורשה והכנסו ל-S.

בכל מקרה מתבצעות שלוש פעולות הכנסה/הוצאה לכל היותר, לכן הסיבוכיות הכוללת היא $3 \cdot O(\log n) = O(\log n)$.

- find_min – החזר את ערך השדה min בשורש הערימה S. הסיבוכיות $O(1)$.

- find_max – החזר את ערך השדה max בשורש הערימה L. הסיבוכיות $O(1)$.

- find_mid – החזר את ערך השורש של הערימה S. הסיבוכיות $O(1)$.
- del_min – בצע חיפוש של המינימום בערימה S: החל מהשורש בכל צומת בדוק באיזה מבניו ערך השדה min קטן יותר ורד לאותו הבן. העלה שנמצא בסוף המסלול הזה הינו המינימום ויש למחוק אותו. ההוצאה תתבצע כמו שמתואר בתרגיל כיתה מס' 10. הסיבוכיות $O(\log n)$.
- del_max – הפעולה מתבצעת באופן סימטרי ל-del_min, אבל בערימה L.
- del_mid – בצע מחיקת שורש מערימה S. הסיבוכיות $O(\log n)$.

בכל אחת מפעולות מחיקה יש לבצע גם עדכון של $balance$ והעברת איברים באופן סימטרי לפעולת הכנסה.

4

א. לכל אחד מהאיברים באוסף ניצור רשומה בעלת שני שדות: val שיכיל את הערך עצמו ו-set לשמירת שם הקבוצה. אוסף הרשומות האלה נכניס למערך A, הדבר יתפוס $O(n)$ מקום.
נשתמש Counting Sort למיון המערך לפי שדות val של הרשומות. המיון ידרוש $O(n+n) = O(n)$ זמן וגם $O(n)$ מקום נוסף.
כעת נפעיל Counting Sort למיון המערך לפי שדות set. גם פה זה יקח $O(n)$ זמן ו-$O(n)$ מקום.
מכיוון שהמיון הזה הוא יציב, נקבל כתוצאה מערך בו כל האיברים מחולקים לקבוצות שלהם ובתוך כל קבוצה הינם ממויינים לפי ערכיהם.

ב. נשתמש ברעיון של ייצוג המספרים בבסיס n (ראה תרגיל כיתה מס' 11).
בייצוג לפי בסיס n כל המספרים מהתחום הנתון יהיו באורך $p+1$ לכל היותר. לכן ניתן יהיה להפעיל עליהם את המיון Radix Sort בסיבוכיות זמן $O((p+1)(n+n)) = O(pn)$.
כדי לעמוד בדרישת סיבוכיות מקום נוסף ולא לקלקל את הקלט יש לבצע העברת המספרים לבסיס החדש בשלבים (נמספרם מ-0 עד p): בשלב ה-i-י "נגזור" ספרה i-ית (בייצוג לפי בסיס n) מכל המספרים, נמיינם לפי הספרה הזו ונעבור לשלב הבא.
כך בכל שלב אנו משתמשים ב-$O(n)$ זכרון נוסף.

5

רעיון כללי: כל איבר x שנמצא במקום "שלו" (המקום שנקבע לו ע"י $h(x)$) אמור להכנס לטבלה בין הראשונים.

א. עבור כל אחד מהאיברים בטבלה נחשב פונקצית הערבול ונשווה אותה עם המיקום של האיבר:

$$\begin{array}{llll} h(36) = 3 \neq 0 & h(26) = 4 \neq 1 & h(11) = 0 \neq 2 & h(12) = 1 \neq 3 \\ h(46) = 2 \neq 4 & h(17) = 6 \neq 5 & h(47) = 3 \neq 6 & h(16) = 5 \neq 7 \\ h(7) = 7 \neq 8 & h(8) = 8 \neq 9 & h(9) = 9 \neq 10 & \end{array}$$

כלומר, אף אחד מהאיברים לא נמצא במקום "שלו". המצב לא יתכן כי לפחות האיבר הראשון בסדרת ההכנסות חייב לתפוס את המקום הנכתב לו ע"י פונקצית ערבול עצמה.

ב. קודם כל נכניס לטבלה את הערכים הנמצאים במקומות "שלהם" (הסדר בינהם לא משנה). בכל צעד הבא נמצא איבר אשר בהכנסתו לטבלה "דורך" רק על המקומות שתפוסים כבר.

סדרת הכנסות אפשרית (קיימים גם פתרונות אחרים):

7, 19, 8, 50, 30, 16, 13, 45, 52, 22, 47