מבני נתונים
תרגיל 3 - פתרונות

מיונים לינאריים

1. הריצו את מיון מנייה על המערך הבא:

$$5, 4, 1, 5, 1, 1, 2, 2, 5, 4, 4$$

הניחו שהחסם העליון לערכי המספרים במערך הוא $k = 6$, והראו את שלבי הריצה של האלגוריתם. אין צורך לרדת לפרטי פרטים, בעיקר ציינו איך נראה מערך המונים אחרי כל שלב ותארו כמה שלבים של כתיבת האברים במערך החדש, יחד עם מה שקורה למערך המונים באותו זמן.

פתרון:

(א) לאחר בניית מערך המונים:

	0	1	2	3	4	5
c	0	3	2	0	3	3

(ב) ואז הוא הופך למערך המייצג את האינדקס האחרי אחרון של כל מספר:

	0	1	2	3	4	5
c	0	3	5	5	8	11

(ג) בסוף ממינים את המערך עצמו, ובמהלך זה מערך המונים קטן כך שלבסוף הוא מצביע על האינדקס הראשון של כל אבר: נראה פה איך המערך b מתמלא תוך כדי:

-

	0	1	2	3	4	5
c	0	3	5	5	8	11

	0	1	2	3	4	5	6	7	8	9	10
b											

	0	1	2	3	4	5
c	0	3	5	5	7	11

	0	1	2	3	4	5	6	7	8	9	10
b								4			

	0	1	2	3	4	5
c	0	3	5	5	6	11

	0	1	2	3	4	5	6	7	8	9	10
b							4	4			

	0	1	2	3	4	5
c	0	3	5	5	6	10

	0	1	2	3	4	5	6	7	8	9	10
b							4	4			5

	0	1	2	3	4	5
c	0	3	4	5	6	10

	0	1	2	3	4	5	6	7	8	9	10
b					2		4	4			5

	0	1	2	3	4	5
c	0	3	3	5	6	10

- וכך הלאה עד ל:
	0	1	2	3	4	5
c	0	0	3	5	5	8

	0	1	2	3	4	5	6	7	8	9	10
b	1	1	1	2	2	4	4	4	5	5	5

2. במיון מנייה, אילו היינו משנים את הקוד של הלולאה השנייה שמכניסה את המספרים חזרה למערך, ובמקום שכתוב:

```
for (int i=a.length-1; i>=0; i--) {
  int value = a[i];
  int index = c[value] - 1;
  b[index] = value;
  c[value]--;
}
```

היינו מורידים את השורה:

```
  c[value]--;
```

האם האלגוריתם היה ממשיך לפעול נכון על קלט כלשהו? על אף קלט? על חלק מהקלטים? הסבירו את תשובותיכם.

פתרון: האלגוריתם היה ממשיך לפעול נכון על כל קלט בו כל הערכים הם שונים. אבל בקלט בו יש שני ערכים זהים, האלגוריתם יכתוב אותם אחד על השני כשהוא כותב את הערכים במערך ולכן הם יופיעו רק פעם אחת. במקומות האחרים בהם היו צריכים להופיע יופיע 0.

3. במיון מנייה:

(א) מה יקרה אם בשלב האחרון בו אנו מכניסים את הערכים מהמערך המקורי למערך החדש b, נרוץ על המערך המקורי מההתחלה לסוף ולא כפי שזה ממומש כרגע, מהסוף להתחלה?

פתרון: אנו נאבד את תכונת היציבות, כי האברים עם ערכים זהים יכנסו בסדר הפוך במערך החדש.

(ב) כרגע, אחרי הלולאה השנייה במיון מנייה, מערך המונים מחזיק במקום ה i את המקום האחרון (פלוס 1) שבו יופיעו האברים עם ערך i. איך נשנה את הקוד כך שמערך המונים יחזיק לא את המקום האחרון אלא את המקום הראשון שבו יופיעו ה i?

פתרון: נוסיף את 3 השורות הבאות לאחר הלולאה השנייה בשיטה:

```
for (int i=k-1; i>0; i--)
  c[i] = c[i-1];
c[0] = 0;
```

(ג) השתמשו בסעיף הקודם וכתבו גרסא חדשה למיון מנייה שתמלא את הערכים חזרה במערך משמאל לימין ולא מימין לשמאל. חשוב שהמיון שאתם כותבים יהיה מיון יציב.

פתרון:

```
void countingSort(int[] a, int k)
{
  int[] c = new int[k];
  for (int i=0; i<a.length; i++)
    c[a[i]]++;
  for (int i=1; i<k; i++)
    c[i] += c[i-1];
  for (int i=k-1; i>0; i--)
    c[i] = c[i-1];
  c[0] = 0;
  int[] b = new int[a.length];
  for (int i=0; i<a.length; i++) {
    int value = a[i];
    b[c[value]] = value;
    c[value]++;
  }
  // now copy b back to a.
}
```

4. הדגימו את מיון דלי על שני המערכים הבאים:

(א)

$$0.79, 0.13, 0.16, 0.64, 0.39, 0.2, 0.89, 0.53, 0.71, 0.42$$

פתרון:

$.0-.1$	$.1-.2$	$.2-.3$	$.3-.4$	$.4-.5$	$.5-.6$	$.6-.7$	$.7-.8$	$.8-.9$	$.9-1$
	$0.13, 0.16$	0.2	0.39	0.42	0.53	0.64	$0.79, 0.71$	0.89	

ואז כל תא ממוין בפני עצמו:

$.0-.1$	$.1-.2$	$.2-.3$	$.3-.4$	$.4-.5$	$.5-.6$	$.6-.7$	$.7-.8$	$.8-.9$	$.9-1$
	$0.13, 0.16$	0.2	0.39	0.42	0.53	0.64	$0.71, 0.79$	0.89	

ומפה שולפים את האברים אחד אחד ומקבלים אותם ממוינים.

(ב)

$$0.63, 0.67, 0.68, 0.66, 0.6, 0.64, 0.65, 0.69, 0.62, 0.61$$

פתרון: כיון שכולם המספרים נופלים בתא אחד, אז ממינים אותם במיון שבחרנו בתור המיון המשני של מיון דלי - מיון הכנסה, ואז הפסדנו את כל מה שמיון דלי נותן.

מהו זמן הריצה כפונקציה של גודל הקלט בשני הסעיפים?

פתרון: בראשון $O(n)$ ובשני $O(n^2)$.

5. כתבו את הקוד שמחשב את מספר הדלי של מספר כאשר הטווח הוא לאו דוקא $[0,1)$ אלא: $[l,r)$. למשל אם הטווח הוא $[1,3)$ ומספר הדליים $n=5$ אז המספר $value=2.01$ צריך להכנס לדלי 2 (הדלי השלישי).

הדרכה: קחו קודם את הערך והיפכו אותו לערך המקביל אילו הטווח כן היה $[0,1)$.

פתרון: דבר ראשון ננרמל למספר בין 0 ו-1: $value' = \frac{value-l}{r-l}$, ואז מספר הדלי הוא כמו בקוד המקורי: $round(value' * n)$.

6. שרטט את מהלכו של מיון בסיס (לא את כל הצעדים, אלא רק את המצב לאחר כל שימוש במיון מניה) על רשימת המלים הבאה:

cow, dog, sea, rug, row, mob, box, tab, bar, ear, tar, dig, big, tea, now, fox

פתרון: מיון אות תחתונה:

sea, tea, mob, tab, dog, rug, dig, big, bar, ear, tar, cow, row, now, box, fox

אות שנייה:

tab, bar, ear, tar, sea, tea, dig, big, mob, dog, cow, row, now, box, fox, rug

ואות ראשונה:

bar, big, box, cow, dig, dog, ear, fox, mob, now, row, rug, sea, tab, tar, tea

בטא את זמן הריצה של האלגוריתם כפונקציה של: מספר המילים n, מספר האותיות בשפה k, ומספר האותיות במילה d.

פתרון: $\Theta(d(n+k))$

7. **בונוס:** נתון מערך עם n מספרים שלמים בטווח $[0,...,n^2-1]$. מצאו אלגוריתם למיון מערך כזה הרץ בזמן $O(n)$.

רמז: חישבו על אלגוריתמי המיון בזמן לינארי שלמדנו.

פתרון: כל מספר בטווח ניתו לחשוב עליו כמספר דו סיפרתי כאשר כל סיפרה היא בטווח $0,...,n-1$. נריץ לכן מיון בסיס, רק שהבסיס שלנו הוא n. לכל

מספר שתי ספרות, ולכן לפי התוצאה של השאלה הקודמת נקבל שזמן הריצה הוא $O(2(n+n)) = O(n)$.

איך מגלים ממספר x בטווח מה הספרה הקטנה ומה הגדולה? כרגיל: הספרה הקטנה היא $x \bmod n$, והגדולה היא $\lfloor \frac{x}{n} \rfloor$.

בעיית הבחירה

1. נדון באלגוריתם הלינארי (בכל מקרה) לפתרון בעית הבחירה.

(א) הציגו דוגמה פשוטה ככל האפשר למערך בן 25 תאים בו חציון החציונים ימוקם במיקום המיטבי. הסבירו היכן ימוקם חציון החציונים, ואילו ערכים ישכנו מכל אחד משני צדדיו.

פתרון:

$$1, 2, 3, 4, 5, 6...$$

כלומר ניקח מערך ממוין. המיון של החמישיות לא יגרום לשום תזוזה. ואז חציון החציונים יהיה מתוך

$$3, 8, 13, 18, 23$$

והוא 13. הוא כמובן ממוקם בדיוק באמצע.

(ב) כמו א', רק שחציון החציונים ימוקם הפעם במקום הגרוע ביותר.
פתרון: נסדר לפי חמישיות:

$$1, 2, 50, 100, 101 \mid 3, 4, 51, 102, 103 \mid 5, 6, 52, 104, 105 \mid 106, 107, 108, 109, 110 \mid 111, 112, 113, 114, 115$$

כל חמישיה כבר ממוינת וחציוני החציונים הם

$$50, 51, 52, 108, 113$$

מבינהם החציון הוא 52 וקטנים ממנו הם רק: $1, 2, 3, 4, 5, 6, 51, 52$.
כדי להבין איך מגיעים למערך הזה, חשבו על המקרה שמותרות חזרות במערך:

$$1, 1, 1, 3, 3 \mid 1, 1, 1, 3, 3 \mid 1, 1, 2, 3, 3 \mid 3, 3, 3, 3, 3 \mid 3, 3, 3, 3, 3$$

פה מקבלים בדיוק את המקרה הגרוע. היחידים שקטנים מהחציון אלה החציונים שקטנים ממנו ואלו שמתחתיהם. בקורס למדנו רק את הגרסא בה כל המספרים שונים, אבל למעשה אין הבדל אמיתי בין גרסא זו והגרסא המלאה.

2. בשאלה זו השתמשו בשיטה select, והניחו שהיא מחזירה את ערך החזרה שלה בלי שהיא משנה את המערך כלל.

(א) תארו אלגוריתם לינארי המדפיס את k האברים הקטנים ביותר במערך לא ממוין.

פתרון:

```
int kth = select(a, 0, a.length, k);
for (int i = 0; i < a.length; i++)
  if (a[i] < kth)
    print a[i];
```

(ב) תארו אלגוריתם לינארי המדפיס את k האברים מהמערך שהכי קרובים למספר נתון. מה זה הכי קרובים? המרחק בין שני מספרים הוא ההפרש בינהם בערך מוחלט. כלומר אנו מחפשים את המספרים שההפרש בינהם ובין הערך שאנו מקבלים הוא הכי קטן. לשם פשטות, הניחו שקבוצת המרחקים בין המספרים השונים אינה מכילה חזרות.

רמז : צרו מערך חדש מהמערך הנתון והשתמשו בו כדי לדעת איזה ערכים מהמערך המקורי הם רלוונטים.

פתרון:

```
int[] b = new int[a.length];
for (int i = 0; i < a.length; i++)
  b[i] = Math.abs(a[i] - value);
int kth = select(b, 0, b.length, k);
for (int i = 0; i < b.length; i++)
  if (b[i] < kth)
    print a[i];
```

(ג) כיתבו אלגוריתם לינארי המוצא את k האברים שהכי קרובים לחציון המערך.

פתרון: נריץ קודם את האלגוריתם של מציאת חציון ואז נריץ את האלגוריתם מהסעיף הקודם עם החציון הזה.

3. באלגוריתם לבעיית הבחירה, אילו היינו עובדים לא עם חמישיות אלא עם שלשות בשלב של מציאת ה־ pivot מה היה יוצא הביטוי הרקורסיבי המבטא את זמן הריצה $T(n)$? אין צורך לפתור את המשוואה הרקורסיבית, אלא רק לכתוב אותה.

פתרון: הקריאה הרקורסיבית הראשונה תהיה על כל האמצעים של השלישיות, ויש $\frac{1}{3}n$ כאלה.

אז צריך לשאול כמה טוב יהיה ה pivot שמצאנו. הוא יותר גדול מ $\frac{1}{6}n$ החציונים שקטנים ממנו, וכל אחד כזה מביא כנדוניה עוד אחד. סך הכל $\frac{2}{6}n = \frac{1}{3}n$. זאת

אומרת שה pivot נמצא איפשהו בטווח $\frac{1}{3}n...\frac{2}{3}n$, ולכן במקרה הגרוע הקריאה הרקורסיבית תהיה על $\frac{2}{3}n$.

סך הכל:

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

ואגב, רק לשם ההשכלה הכללית, פתרון נוסחא זו יתן $T(n) = \Theta(n \log n)$.