Sorting - מיון

<u>משפט</u>: כל אלגוריתם מיון מבוסס השוואות הממיין n מפתחות כלשהם מבצע לכל הפחות $\Omega(n \log n)$ פעולות השוואה במקרה הגרוע ובמקרה הממוצע.

לפעמים, אם ידוע מידע נוסף על הקלט, ניתן למיין בסיבוכיות זמן טובה יותר.

Counting Sort

אם n המספרים לקוחים מהטווח $[1,k]$, ניתן למיין אותם בזמן $O(n+k)$.

- עבור $k = O(n)$ נקבל מיון בזמן ליניארי.

Counting Sort הינו אלגוריתם מיון **יציב**:

הגדרה: אלגוריתם מיון הוא יציב אם הוא שומר על הסדר היחסי בין איברים בעלי ערכים זהים.

3 1 3 0 5 3 → 0 1 3 3 3 5

Counting Sort

בהינתן מערך A בגודל n שאיבריו לקוחים מהטווח $[1,k]$:

	1	2	3	4	5	6	7	8
A	6	1	2	1	2	5	6	2

1. <u>נקצה מערך C בגודל k שיספור כמה פעמים מופיע כל איבר $i \in [1,k]$:</u>

	1	2	3	4	5	6
C	2	3	0	0	1	2

```
for (int i=1; i <= k; i++) C[i] ← 0;
for (int i=1; i <= n; i++) C[A[i]]++;
```

2. <u>נמצא לכל איבר $i \in [1,k]$ את האינדקס האחרון בסדרה הממויינת שבו מופיע i.</u>

	1	2	3	4	5	6
C	2	5	5	5	6	8

```
for (int i=2; i <= k; i++)
    C[i] ← C[i-1] + C[i];
```

Counting Sort

3. נעבור על האיברים מהסוף להתחלה ונמקם אותם במערך הפלט B.

```
for (int i=n; i >= 1; i--)
    B[C[A[i]]] ← A[i];
    C[A[i]]--;
```

	1	2	3	4	5	6	7	8
A	6	1	2	1	2	5	6	2

	1	2	3	4	5	6	7	8
B								

	1	2	3	4	5	6
C	2	5	5	5	6	8

	1	2	3	4	5	6	7	8
B					2			

	1	2	3	4	5	6
C	2	4	5	5	6	8

	1	2	3	4	5	6	7	8
B					2			6

	1	2	3	4	5	6
C	2	4	5	5	6	7

	1	2	3	4	5	6	7	8
B					2	5		6

	1	2	3	4	5	6
C	2	4	5	5	5	7

	1	2	3	4	5	6	7	8
B				2	2	5		6

	1	2	3	4	5	6
C	2	3	5	5	5	7

Counting Sort

3. נעבור על האיברים מהסוף להתחלה ונמקם אותם במערך הפלט B.

```
for (int i=n; i >= 1; i--)
    B[C[A[i]]] ← A[i];
    C[A[i]]--;
```

	1	2	3	4	5	6	7	8
A	6	1	2	1	2	5	6	2

	1	2	3	4	5	6	7	8
B		1		2	2	5		6

	1	2	3	4	5	6
C	1	3	5	5	5	7

	1	2	3	4	5	6	7	8
B		1	2	2	2	5		6

	1	2	3	4	5	6
C	1	2	5	5	5	7

	1	2	3	4	5	6	7	8
B	1	1	2	2	2	5		6

	1	2	3	4	5	6
C	0	2	5	5	5	7

	1	2	3	4	5	6	7	8
B	1	1	2	2	2	5	6	6

מיון ב-O(n)?

מיון Counting Sort לוקח $O(n+k)$ זמן, כאשר המספרים לקוחים מהטווח $[1,k]$.

נסתכל על האלגוריתם הבא:

- נעבור על המערך בזמן $O(n)$ ונמצא את האיבר המקסימלי בו. נסמן איבר זה ב-k.
- נשתמש ב-Counting Sort על מנת למיין את המערך.
- היות ו-k הוא קבוע, נקבל מיון בזמן $O(n)$.

איפה הטעות?

האיבר המקסימלי במערך יכול להיות 2^n.

Counting Sort במצב זה ייקח $O(n+2^n)=O(2^n)$ זמן.