ADATSZERKEZETEK ÉS ALGORITMUSOK

Leszámláló rendező

Leszámláló rendezés

- Tegyük fel, hogy van n db bemeneti elem, s ezek mindegyike 1 és k közötti egész szám
- Az alapötlet: meghatározzuk minden egyes x bemeneti elemre azoknak az elemeknek a számát, amelyek kisebbek, mint az x
- Ezután x-et közvetlenül a saját pozíciójára tudom helyezni
- Legyen a bemenet az $A[1..n]$ tömb, a kimenet a $B[1..n]$ tömb
  - Mindkettő hossza: $\mathrm{hossz}[A] = \mathrm{hossz}[B] = n$
- Szükség van még egy $C[1..k]$ tömbre átmeneti munkaterületként

Leszámláló rendezés

1. Végigmegyünk az A-n, és ha egy elem értéke i, akkor megnöveljük $C[i]$ értékét eggyel.
2. Minden i-re $1..k$ között meghatározzuk, hogy hány olyan bemeneti elem van, amelyiknek az értéke $\leq$ i (összegzés C-n)
3. Minden i-re $n..1$ között $A[i]$-t betesszük B megfelelő pozíciójába – ezt a C-ből állapítjuk meg

- Ha betettük, akkor a $C[A[i]]$ értékét csökkentjük, így a következő vele egyenlő elem már elé kerül, vagyis így stabil lesz a rendezés, az egyenlő elemeknél megtartja az eredeti sorrendet

Leszámláló rendezés

- Az A tömb elemeit leszámláljuk a C tömbbe
  - A C-ben az i helyen az i-vel egyenlő elemek száma szerepel
  - Végigmegyünk az A-n és ha egy elem értéke i, akkor a $C[i]$ értéket megnöveljük
- A

3	6	4	1	3	4	1	4

- C

2	0	2	3	0	1

Leszámláló rendezés

- Minden i-re $1 \dots k$ között meghatározzuk, hogy hány olyan bemeneti elem van, amelyiknek az értéke $\leq i$
(összegzés C-n)
- A

3	6	4	1	3	4	1	4

- C

2	2	4	7	7	8

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C

3	6	4	1	3	4	1	4

- B

2	2	4	7	7	8

						4	

Leszámláló rendezés

- Minden i-re $n \ldots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A

3	6	4	1	3	4	1	4

- C

2	2	4	6	7	8

- B

	1					4	

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C

3	6	4	1	3	4	1	4

- B

1	2	4	6	7	8

	1				4	4	

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C

3	6	4	1	3	4	1	4

- B

1	2	4	5	7	8

	1		3		4	4	

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C
- B

3	6	4	1	3	4	1	4

1	2	3	5	7	8

1	1		3		4	4	

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C

3	6	4	1	3	4	1	4

- B

0	2	3	5	7	8

1	1		3	4	4	4	

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C

3	6	4	1	3	4	1	4

- B

0	2	3	4	7	8

1	1		3	4	4	4	6

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C

3	6	4	1	3	4	1	4

- B

0	2	3	4	7	7

1	1	3	3	4	4	4	6

Leszámláló rendezés

- Minden i-re $n \dots 1$ között $A[i]$-t betesszük B megfelelő pozíciójába - ezt a C-ből állapítjuk meg.
- A
- C

3	6	4	1	3	4	1	4

- B

0	2	2	4	7	7

1	1	3	3	4	4	4	6

Leszámláló rendezés

Az algoritmus pszeudokódja:

```
for i←1 to k do
   C[i]←0
for i←1 to hossz(A) do
   C[A[i]]←C[A[i]]+1
for i←2 to k do
   C[i]←C[i]+C[i-1]
for i←hossz(A) downto 1 do
   B[C[A[i]]]←A[i]
   C[A[i]]←C[A[i]]-1
```

← C-ben az i-vel egyenlő elemek száma

← C-ben az i-nél kisebb, vagy egyenlő elemek száma

Leszámláló rendezés

- Futási idő:
  - 1. for ciklus: $\Theta(k)$
  - 2. for ciklus: $\Theta(n)$
  - 3. for ciklus: $\Theta(k)$
  - 4. for ciklus: $\Theta(n)$
- Így a teljes időigény: $\Theta(k+n)$
  - Ha $k = \Theta(n)$, akkor a rendezés futási ideje $\Theta(n)$!
- Ez nem összehasonlító rendezés
  - A helyigénye viszont nagyobb ☹

Edényrendezés

Következő téma