Counting sort - Sortowanie przez zliczanie

sobota, 15 czerwca 2024 17:21

Mamy liczby z przedziału od 1 do k

tworzymy tablicę w której będziemy zliczać liczbę wystąpień każdej z liczb o wielkości k

Przykład: 1 4 1 2 7 5 2 $k = 9$

C:

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

teraz robimy sumę "zbiorową" na kolejnych komórkach:

C:

1	2	3	4	5	6	7	8	9
2	4	4	5	6	6	7	7	7

teraz w tablicy B o wymiarze n = liczbie elementów sortowanych wstawiamy dane

B:

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

```
procedure CountingSort (A[1...k], k, B[1...n])
    for i=1 to k    C[i] ← 0
    for j=1 to n    C[A[j]] ← C[A[j]] + 1
    for i=2 to n    C[i] ← C[i] + C[i-1]
    for j = n down to 1 do  B[C[A[j]]] ← A[j]
                            C[A[j]] ← C[A[j]] - 1
```

w C[A[j]] trzymamy zliczone wartości

w A[j] mamy te liczby

Złożoność czasowa: $\Theta(n+k)$

Bucket sort - sortowanie kubełkowe

sobota, 15 czerwca 2024 17:54

Mamy ciąg liczb nieujemnych $\nearrow A[1 \dots n]$ z przedziału $[0,1)$

Dzielimy przedział $[0,1)$ na n kubełków jednakowej długości

Umieszczamy liczby w odpowiednich kubełkach → sortujemy kubełki → łączymy je

tablica kubełków

```
bucket-sort (A[1.... n])
    for i=1 to n   B[i] ← 0
    for i=1 to n   dołącz A[i] do listy B[⌊n·A[i]⌋]
    for i=1 to n   posortuj select-sortem listy B[i]
    połącz listy B[1],..., B[n]
```

złożoność : $\Theta(n)$

??! dowód typu sraka :(((

Radix sort

sobota, 15 czerwca 2024 18:25

Wszystkie ciągi są o długości d

```
procedure radix-sort($A_1, \dots, A_n$)
    for $i = d$ down to 1 do
        posortuj stabilnie ciągi wg $i$-tego elementu
```

Koszt: $O((n+k)\cdot d)$