

Сортировки за линейное время

## Сортировки за линейное время

Сортировка подсчетом = Counting Sort

Вход:  $n$  целых чисел из промежутка от 0 до  $k$

Основная идея: для каждого элемента определить число элементов, которые его меньше

## Сортировка подсчетом

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

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

*Handwritten annotations:*  
Under array C:  $+1$  under index 0,  $+1$  under index 1,  $+1$  under index 3,  $+1$  under index 4.

- 1 Пусть  $C[0..k]$  — новый массив
- 2 **for**  $i = 0$  **to**  $k$
- 3      $C[i] = 0$

## Сортировка подсчетом

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

	0	1	2	3	4	5
C	2	2	0	3	1	1
	+2	+4	+4	+7	+8	

```
4  for j = 1 to A.length
5      C[A[j]] = C[A[j]] + 1
6  // Сейчас C[i] содержит количество элементов, равных i.
```

## Сортировка подсчетом

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

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

Handwritten diagram showing the counting sort process. It consists of a horizontal line with vertical dividers. Above the line, indices 2, 4, 7, 9, and 9 are written. Below the line, the values 0, 1, 2, 3, 4, and 5 are written, representing the cumulative counts for each index.

```
7 for  $i = 1$  to  $k$ 
8    $C[i] = C[i] + C[i - 1]$ 
9 // Сейчас  $C[i]$  содержит количество элементов, не превышающих  $i$ .
```

## Сортировка подсчетом

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

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

```
10 for j = A.length downto 1
11     B[C[A[j]]] = A[j]
12     C[A[j]] = C[A[j]] - 1
```

	1	2	3	4	5	6	7	8	9
B							3		

## Сортировка подсчетом

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

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

```
10  for  $j = A.length$  downto 1
11       $B[C[A[j]]] = A[j]$ 
12       $C[A[j]] = C[A[j]] - 1$ 
```

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

## Сортировка подсчетом

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

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

```
10 for  $j = A.length$  downto 1
11      $B[C[A[j]]] = A[j]$ 
12      $C[A[j]] = C[A[j]] - 1$ 
```

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



## Сортировка подсчетом

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

	0	1	2	3	4	5
C	2	3	4	5	8	9

```
10 for  $j = A.length$  downto 1
11      $B[C[A[j]]] = A[j]$ 
12      $C[A[j]] = C[A[j]] - 1$ 
```

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

## Сортировка подсчетом

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

	0	1	2	3	4	5
C	0	2	4	4	7	8

```
10 for  $j = A.length$  downto 1
11      $B[C[A[j]]] = A[j]$ 
12      $C[A[j]] = C[A[j]] - 1$ 
```

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

COUNTING-SORT( $A, B, k$ )

```

1 Пусть C[0..k] — новый массив
2 for i = 0 to k      ]  $\Theta(k)$ 
3   C[i] = 0
4 for j = 1 to A.length ]  $\Theta(n)$ 
5   C[A[j]] = C[A[j]] + 1
6 // Сейчас C[i] содержит количество элементов, равных i.
7 for i = 1 to k      ]  $\Theta(n)$ 
8   C[i] = C[i] + C[i - 1]
9 // Сейчас C[i] содержит количество элементов, не превышающих i.
10 for j = A.length downto 1 ] 
11   B[C[A[j]]] = A[j]      ]  $\Theta(n)$ 
12   C[A[j]] = C[A[j]] - 1

```

**Устойчив!** - элементы с одинаковым значением находятся в выходном массиве в том же относительном порядке, что и во входном

## Сортировки за линейное время

Поразрядная сортировка = Radix Sort

Вход: строки одинаковой длины

Основная идея: сортировать от младшего разряда к старшему с помощью устойчивой сортировки

## Цифровая сортировка

RADIX-SORT( $A, d$ )

1 **for**  $i = 1$  **to**  $d$

2     Выполнить устойчивую сортировку массива  $A$  по цифре  $i$

BAR

BOX

BIG

ROW

EAR

RUG

COW

TAR

TAN

## Цифровая сортировка

**RADIX-SORT**( $A, d$ )

1 **for**  $i = 1$  **to**  $d$

2     Выполнить устойчивую сортировку массива  $A$  по цифре  $i$

BAR

BOX

BIG

ROW

EAR

RUG

COW

TAR

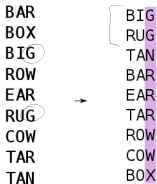
TAN

## Цифровая сортировка

**RADIX-SORT**( $A, d$ )

1 **for**  $i = 1$  **to**  $d$

2     Выполнить устойчивую сортировку массива  $A$  по цифре  $i$

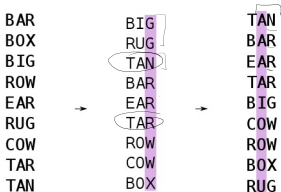


## Цифровая сортировка

RADIX-SORT( $A, d$ )

1 **for**  $i = 1$  **to**  $d$

2     Выполнить устойчивую сортировку массива  $A$  по цифре  $i$





# Цифровая сортировка

RADIX-SORT( $A, d$ )

$$O(d(n+k))$$

1 for  $i = 1$  to  $d$

2 Выполнить устойчивую сортировку массива  $A$  по цифре  $i$

←  
3 2 1

BAR  
BOX  
BIG  
ROW  
EAR  
RUG  
COW  
TAR  
TAN

→

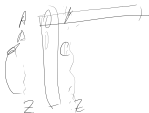
BIG  
RUG  
TAN  
BAR  
EAR  
TAR  
ROW  
COW  
BOX

→

TAN  
BAR  
EAR  
TAR  
BIG  
COW  
ROW  
BOX  
RUG

→

BAR  
BIG  
BOX  
COW  
EAR  
ROW  
RUG  
TAN  
TAR



Название	Время	Доп. память	Ограничения	Устойч.
Выбором, вставкой, пузырьком				
Слиянием				
Быстрая				
Подсчетом				
Поразрядная				
Карманная				
Кучей (итеративная)				
Кучей (рекурсивная)				