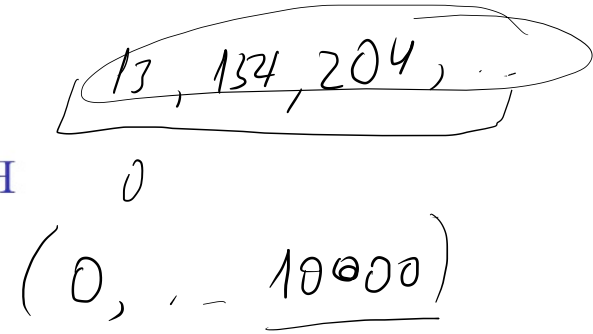
Сортировки за линейное время

Сортировки за линейное время

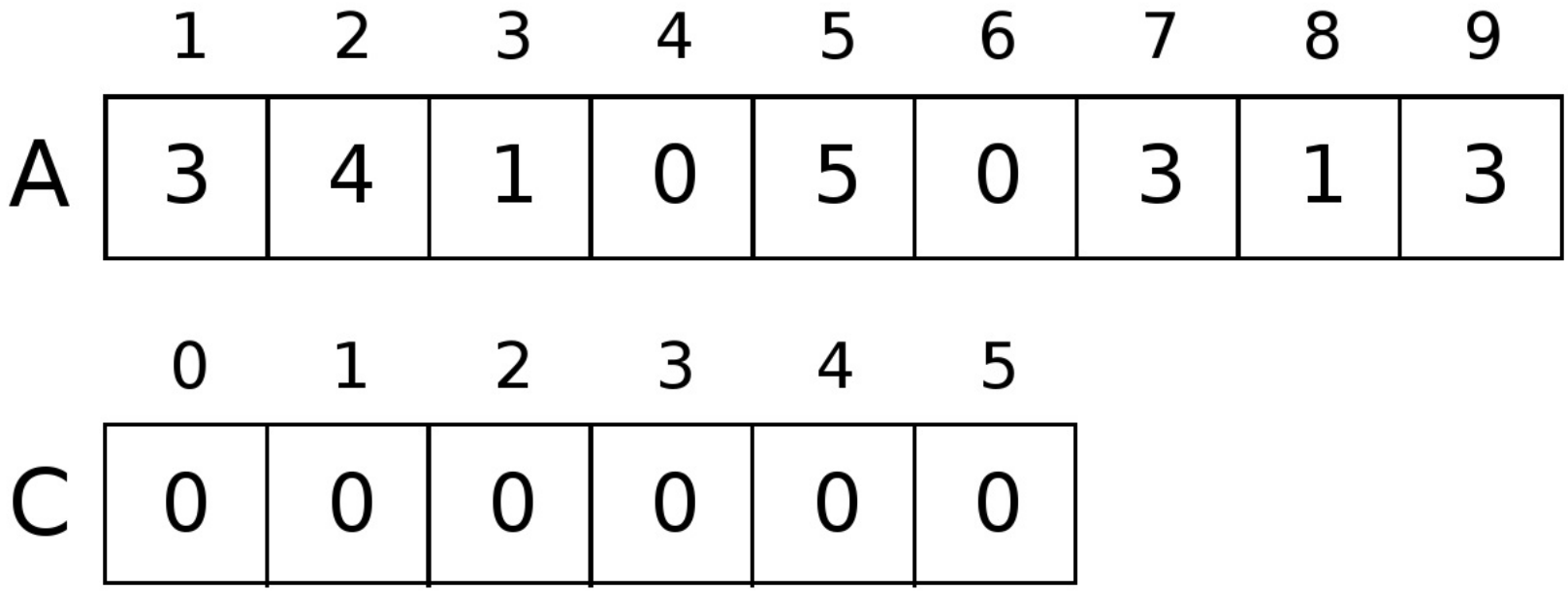


Сортировка подсчетом = Counting Sort

Вход: n целых чисел из промежутка от 0 до k

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

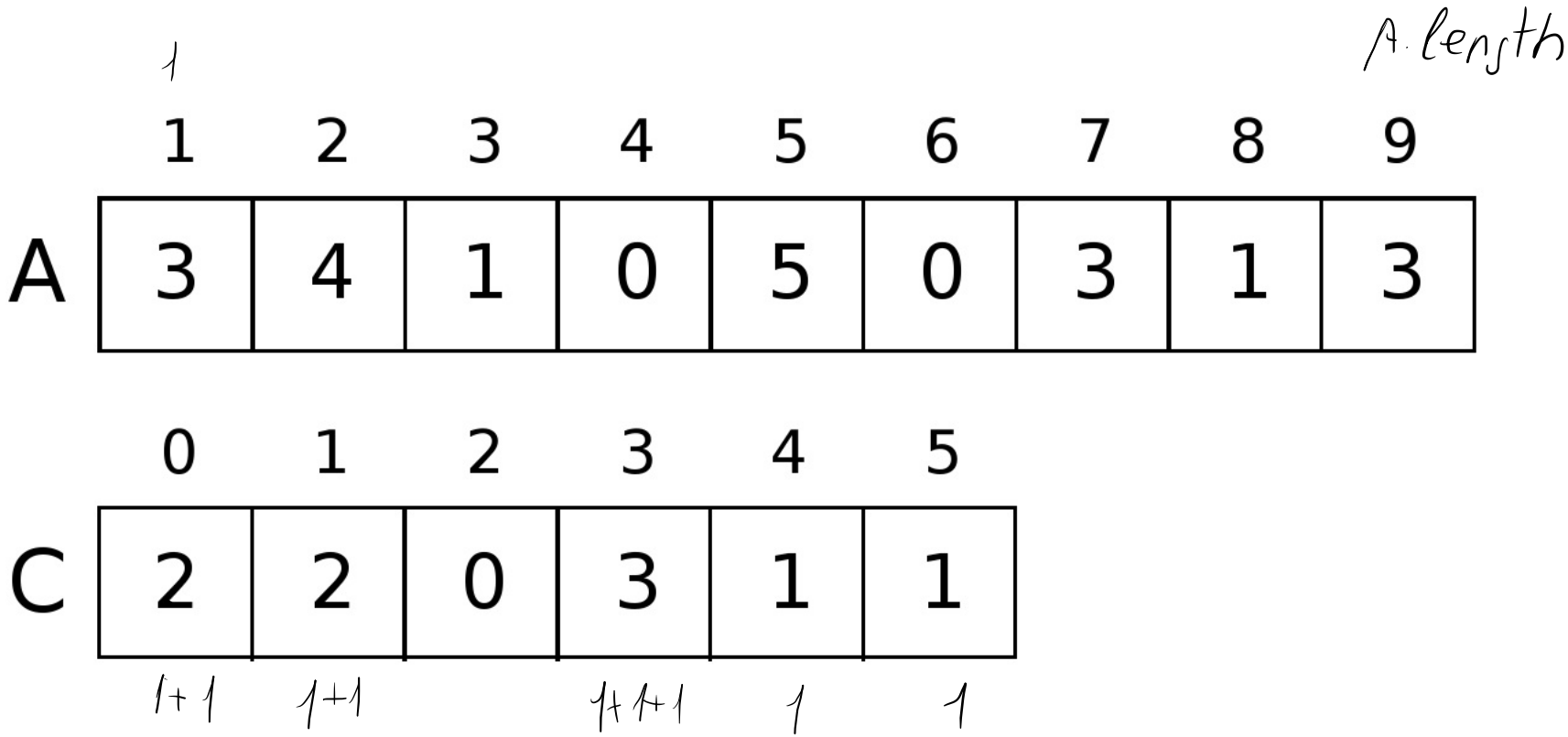
Сортировка подсчетом



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

$\Theta(k)$

Сортировка подсчетом




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

$\Theta(n)$

Сортировка подсчетом

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

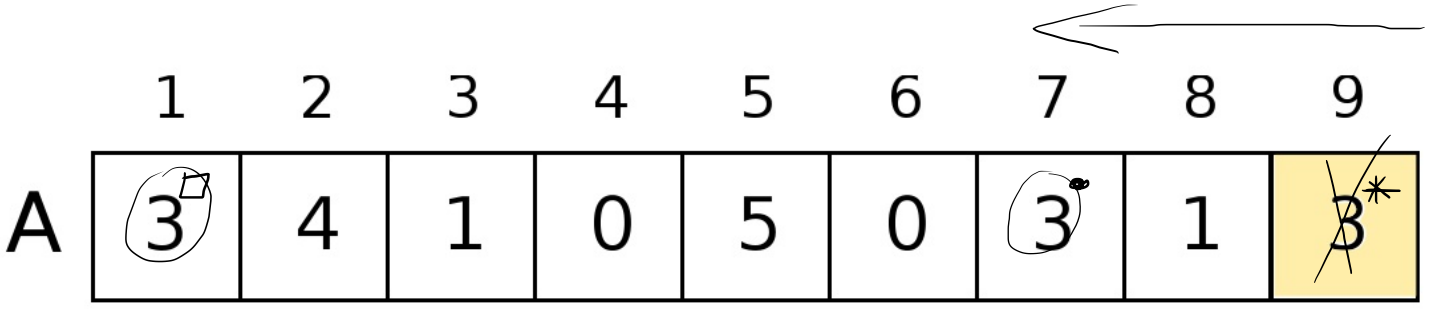
	0	1	2	3	4	5
C	2	4	4	7	8	9

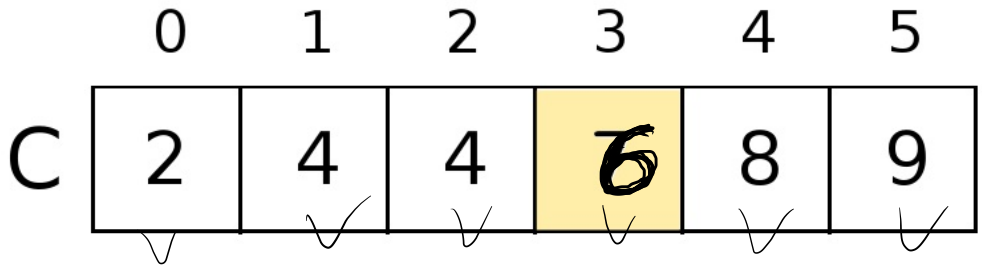
2 2 0 3 1 1
+2 4 +4 8 9

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

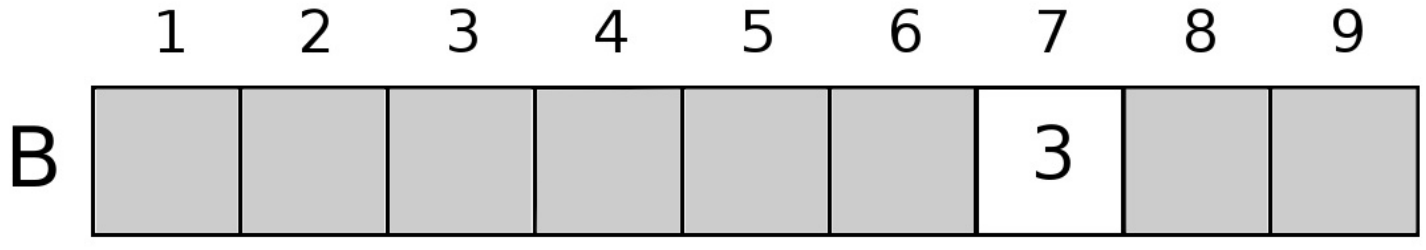
$\Theta(k)$

Сортировка подсчетом

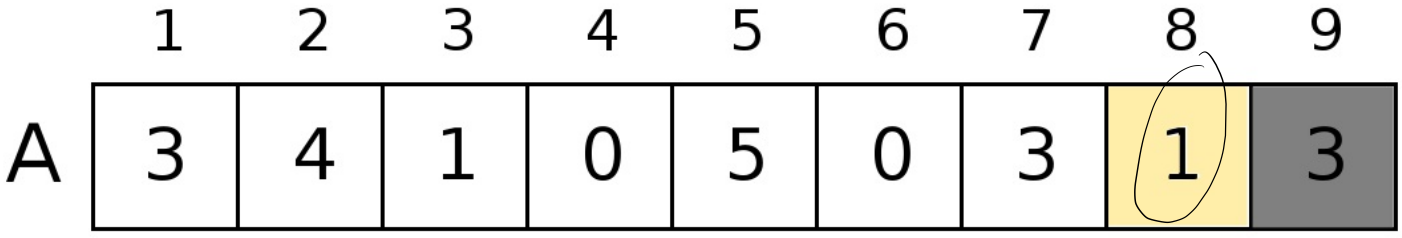


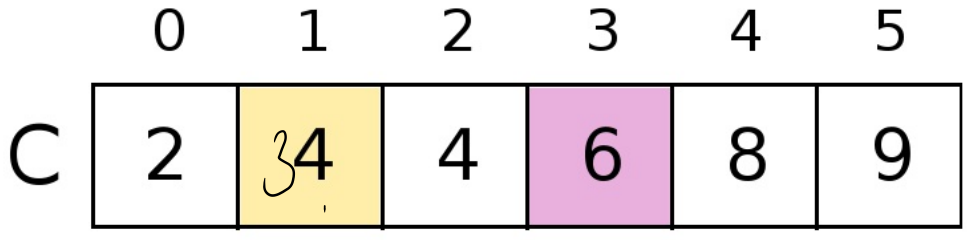



```
10  for j = A.length downto 1
11      B[C[A[j]]] = A[j]
12      C[A[j]] = C[A[j]] − 1
```

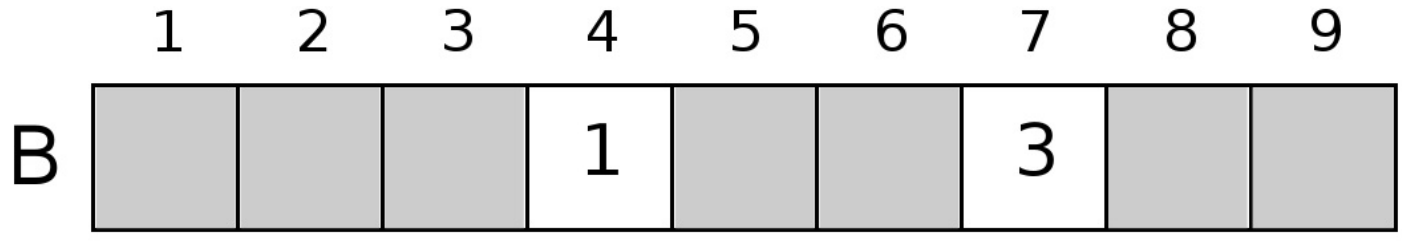


Сортировка подсчетом

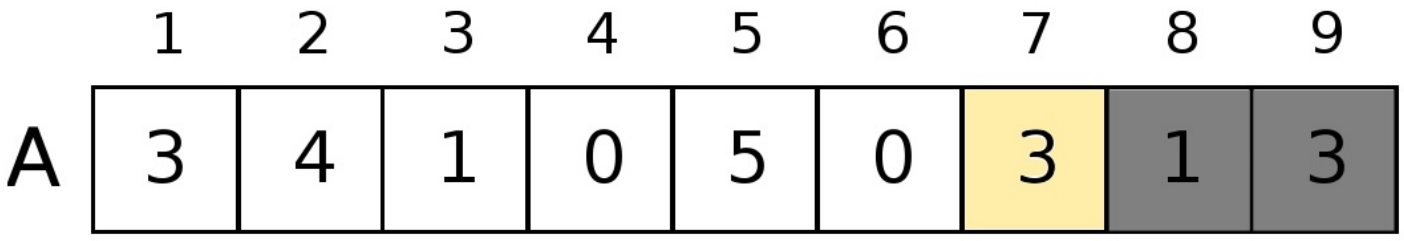


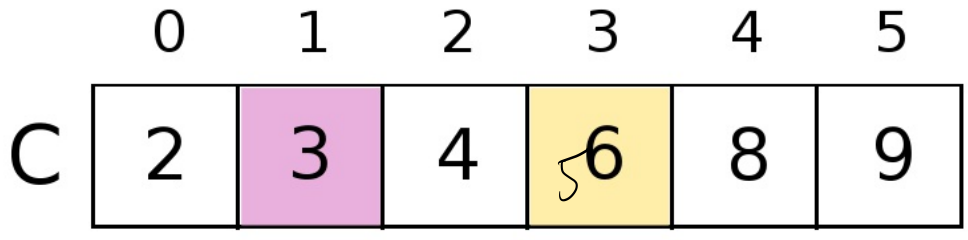



```
10  for j = A.length downto 1
11      B[C[A[j]]] = A[j]
12      C[A[j]] = C[A[j]] − 1
```

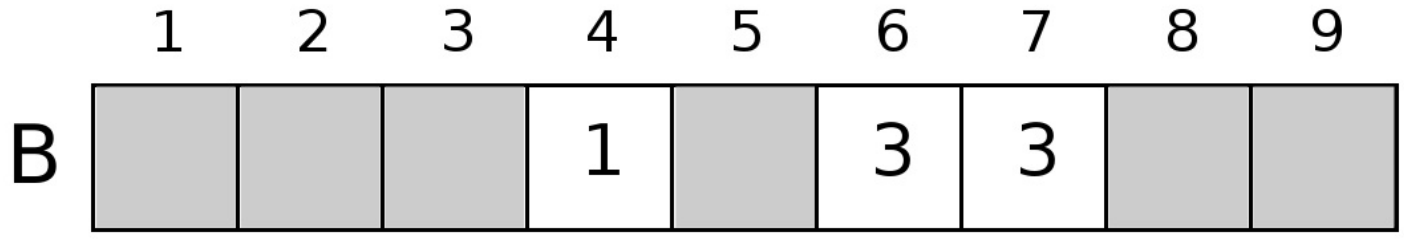


Сортировка подсчетом

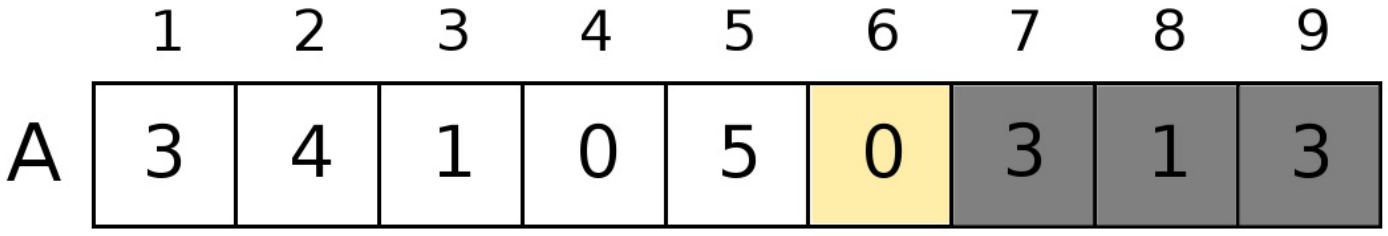


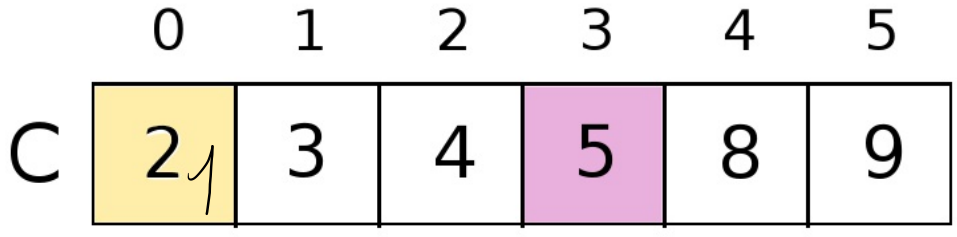



```
10  for j = A.length downto 1
11      B[C[A[j]]] = A[j]
12      C[A[j]] = C[A[j]] − 1
```

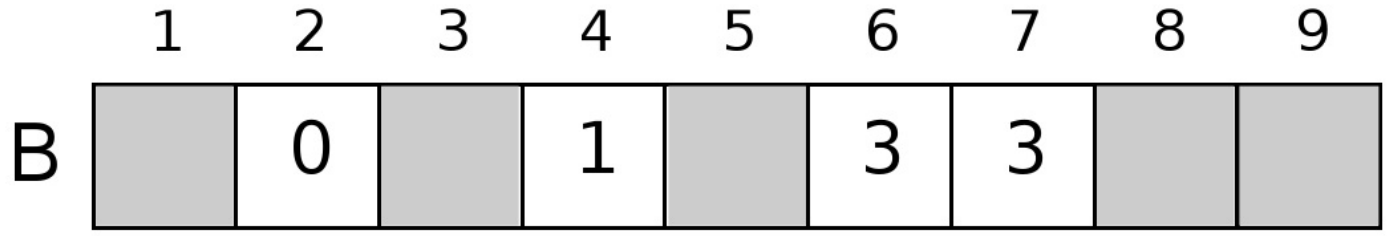


Сортировка подсчетом






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



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

$\Theta(n)$

```
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

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
```

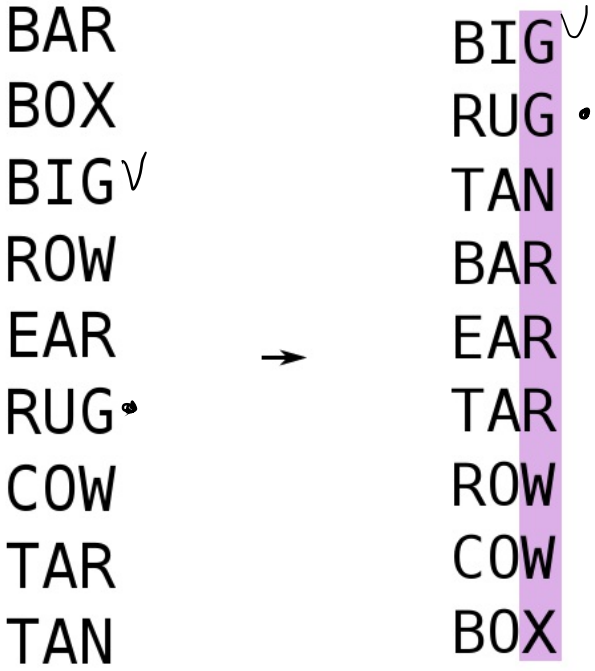
BAR
BOX
BIG
ROW
EAR
RUG
COW
TAR
TAN

Цифровая сортировка

RADIX-SORT(A, d)

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

2 Выполнить устойчивую сортировку массива A по цифре i

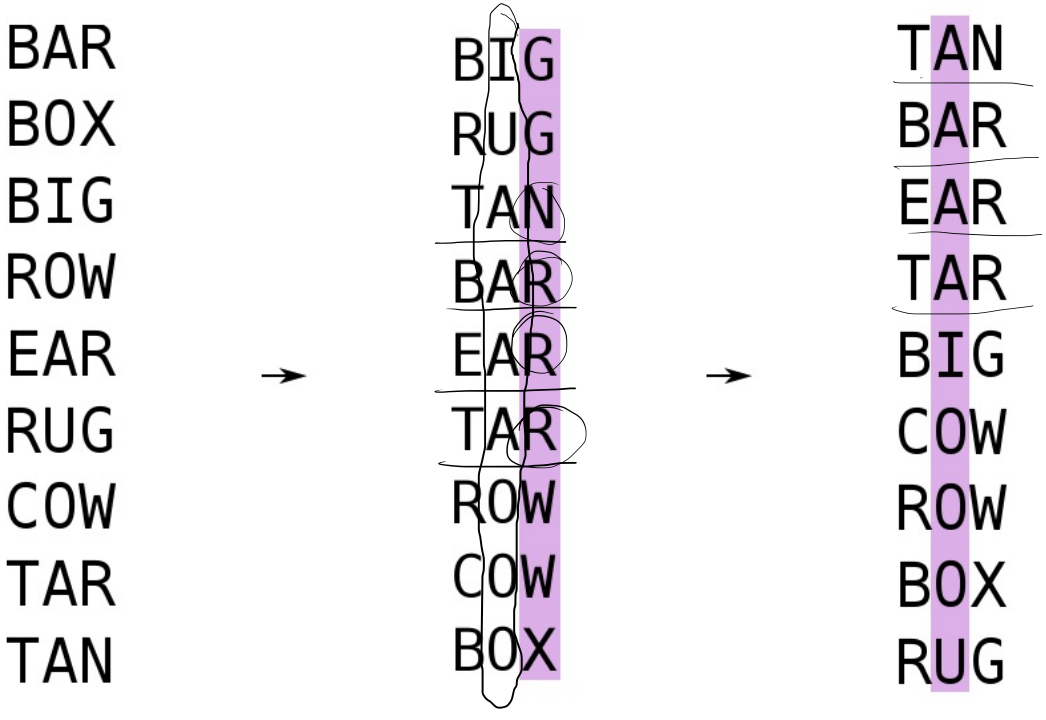


Цифровая сортировка

RADIX-SORT(A, d)

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

2 Выполнить устойчивую сортировку массива A по цифре i

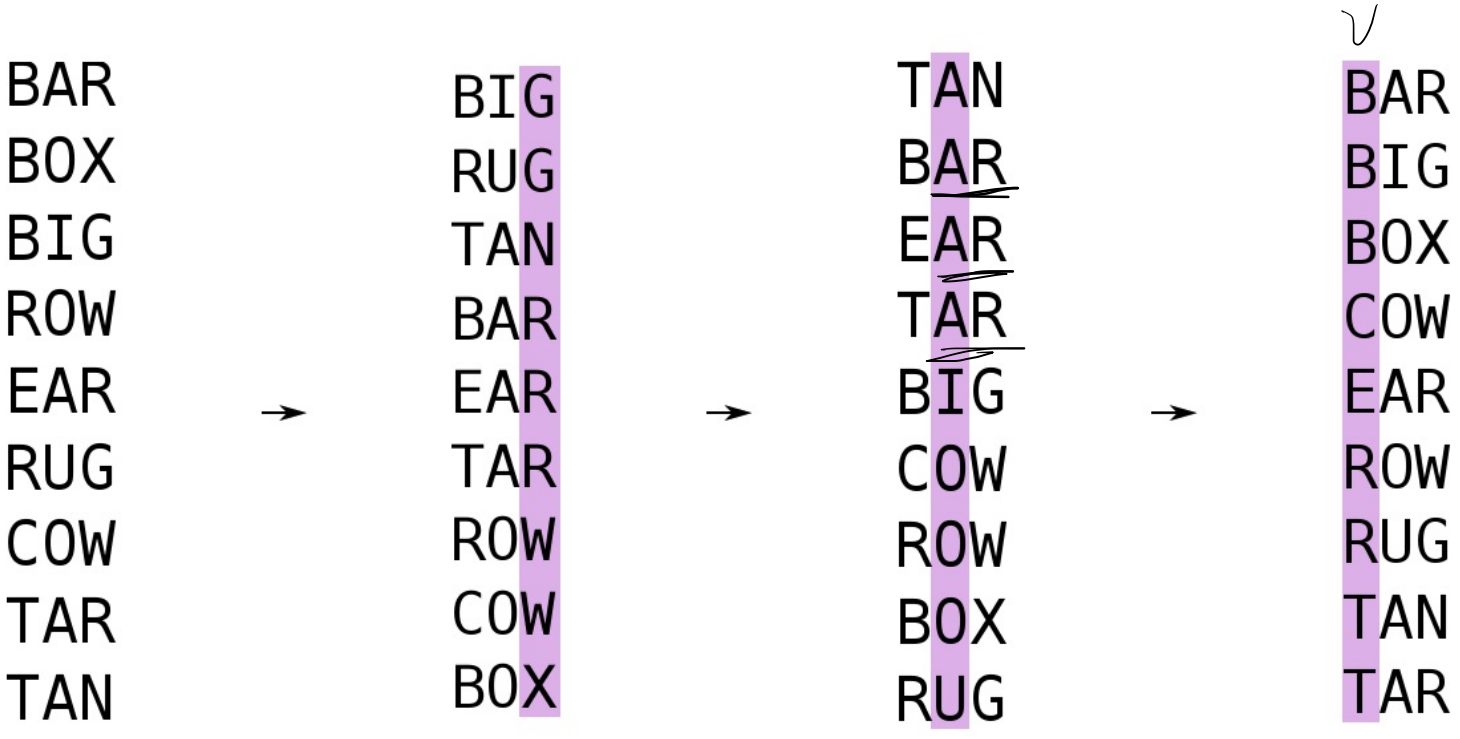


Цифровая сортировка

RADIX-SORT(A, d)

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

2 Выполнить устойчивую сортировку массива A по цифре i



Название	Время	Доп. память	Ограничения	Устойч.
Выбором, вставкой, пузырьком	$O(n^2)$	$O(1)$	—	+
Слиянием	$O(n \log n)$	$O(n)$	—	+
Быстрая	$O(n \log n)$	$O(\log n)$		—
Подсчетом	$O(n+k)$	$O(k)$	целые числа $(0, k)$	+
Поразрядная	$O(d(n+k))$	$O(k)$	d - число разрядов	+
Карманная				
Кучей	$O(n \log n)$	$O(1)$	—	—

C