COUNTING SORT

CS340

Counting Sort

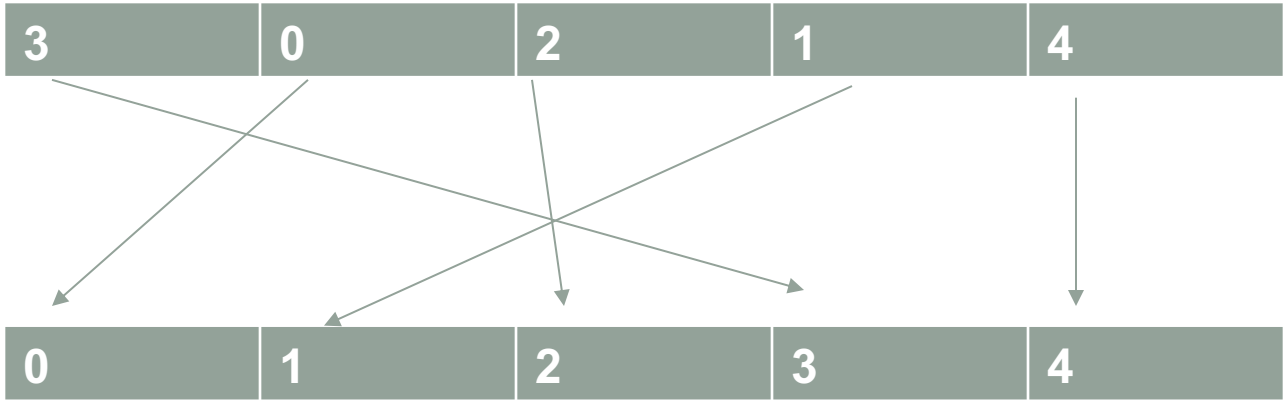
Assumes each of n input elements is an integer in the range 0 to k.

Counting Sort

- Uses 3 arrays:
  - Input array A (size A.length)
  - B[ ] holds the sorted output (size A.length)
  - C[ ] provides temporary working storage (size k+1)

Counting Sort

- The general idea:



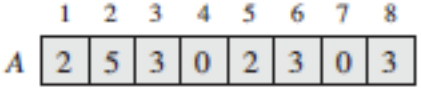
- Make an array of size k+1, put each number at its index.
- What could go wrong?
- This will turn out to be an incredibly powerful idea.

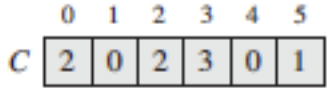
Counting Sort

Elements in A[] <= 5; so k=5

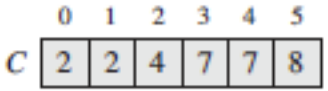

```
COUNTING-SORT(A, B, k)
 1  let C[0..k] be a new array
 2  for i = 0 to k
 3      C[i] = 0
 4  for j = 1 to A.length
 5      C[A[j]] = C[A[j]] + 1
 6  // C[i] now contains the number of elements equal to i.
 7  for i = 1 to k
 8      C[i] = C[i] + C[i − 1]
 9  // C[i] now contains the number of elements less than or equal to i.
10  for j = A.length downto 1
11      B[C[A[j]]] = A[j]
12      C[A[j]] = C[A[j]] − 1
```

Counting Sort





(a)

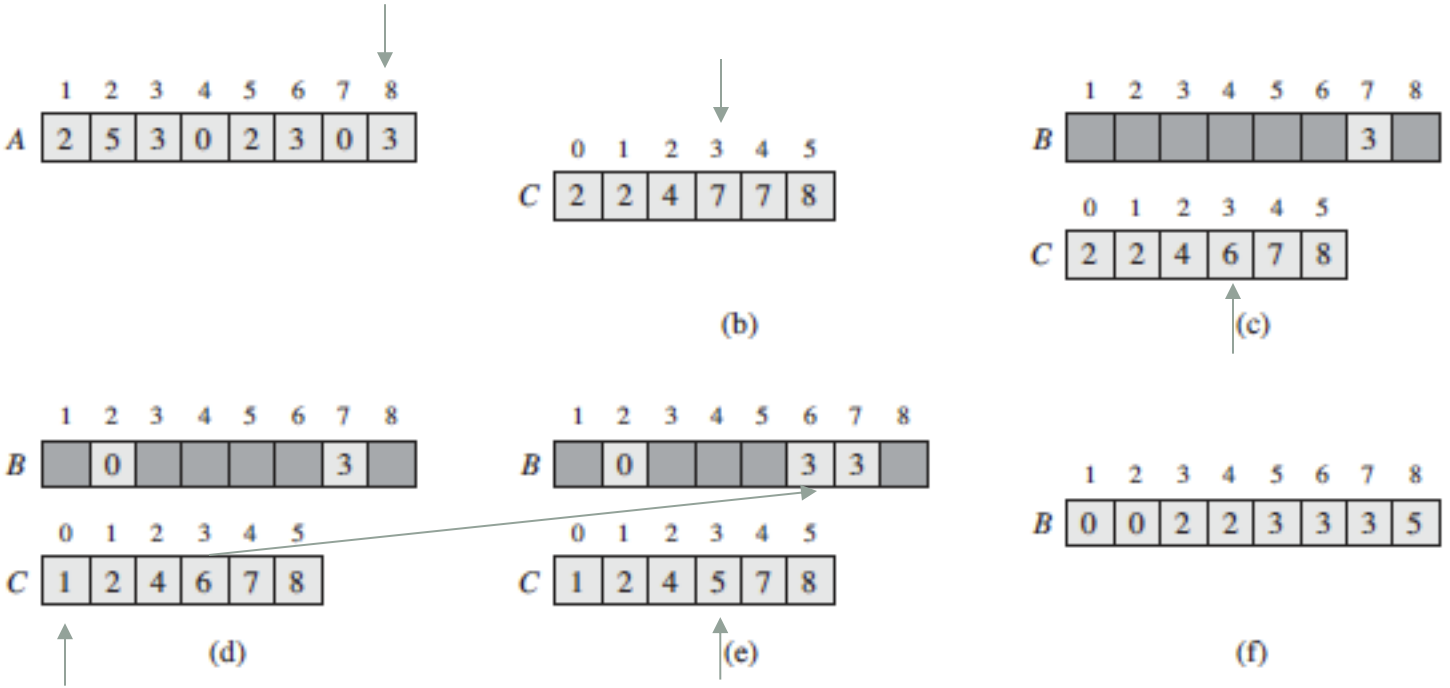


(b)

```
COUNTING-SORT(A, B, k)
 1  let C[0..k] be a new array
 2  for i = 0 to k
 3      C[i] = 0
 4  for j = 1 to A.length
 5      C[A[j]] = C[A[j]] + 1
 6  // C[i] now contains the number of elements equal to i.
 7  for i = 1 to k
 8      C[i] = C[i] + C[i − 1]
 9  // C[i] now contains the number of elements less than or equal to i.
10  for j = A.length downto 1
11      B[C[A[j]]] = A[j]
12      C[A[j]] = C[A[j]] − 1
```

Counting Sort

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



Counting Sort – Time Complexity

- What is the time complexity?
- What's different about this from other sorts?

```
COUNTING-SORT(A, B, k)
1  let C[0..k] be a new array                          Θ(k)
2  for i = 0 to k                                      Θ(k)
3      C[i] = 0
4  for j = 1 to A.length                               Θ(n)
5      C[A[j]] = C[A[j]] + 1                           Θ(n)
6  // C[i] now contains the number of elements equal to i.
7  for i = 1 to k                                      Θ(k)
8      C[i] = C[i] + C[i − 1]                          Θ(k)
9  // C[i] now contains the number of elements less than or equal to i.
10 for j = A.length downto 1                           Θ(n)
11     B[C[A[j]]] = A[j]                               Θ(n)
12     C[A[j]] = C[A[j]] − 1                           Θ(n)
```

More on Counting Sort

- It is *stable*
  - numbers with the same value appear in the output array in the same order as they do in the input array.
  - It breaks ties between two numbers by the rule that whichever number appears first in the input array appears first in the output array.

What are problems with Counting Sort?

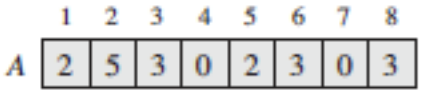
- ???

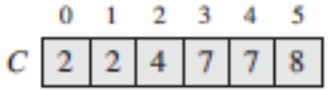
Fix Counting Sort

- Assumes integers in the range 0-k
- Can it be adjusted to accommodate any range of numbers?

Interview Questions

- Describe an algorithm that, given n integers in the range 0 to *k*, preprocesses its input and then answers any query about how many of the *n* integers fall into a range [*a*..*b*] in *O*(1) time. Your algorithm should use Θ(*n*+*k*) preprocessing time.





(b)