Radix Sort

Idea:

- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1.* Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins

d = 3 **a:**

0	329
1	457
2	657
3	839
4	436
5	720
6	355

bins:

0

1

2

3

4

5

6

7

8

9

Radix Sort

Idea:
- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1.* Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins

d = 3 *j = 2*

a:

0	329
1	457
2	657
3	839
4	436
5	720
6	355

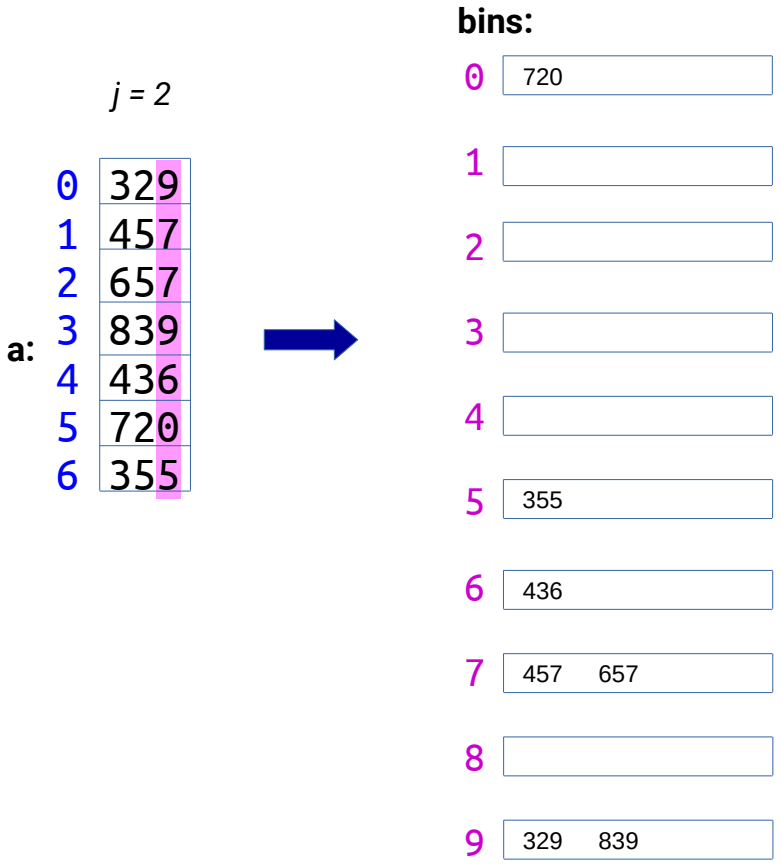
bins:

0
1
2
3
4
5
6
7
8
9

Radix Sort

Idea:

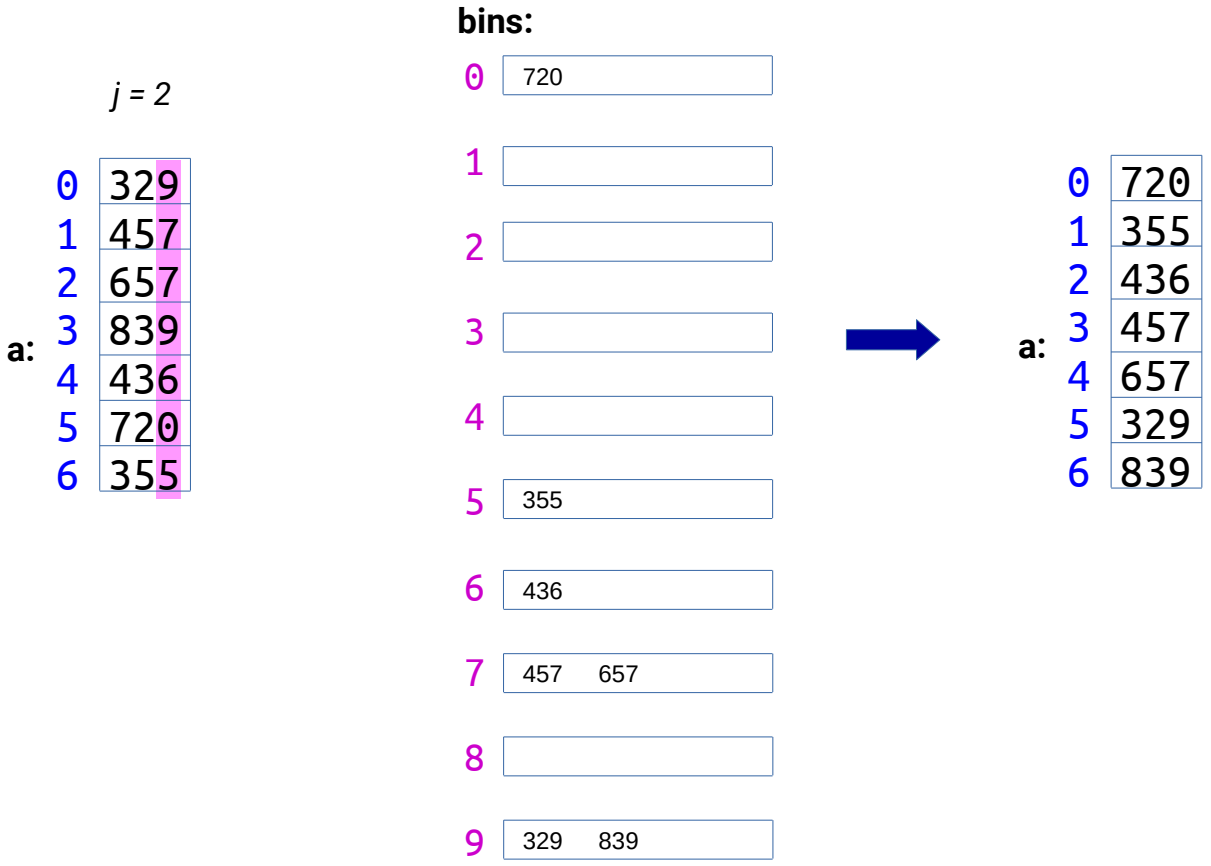
- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1.* Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins



Radix Sort

Idea:

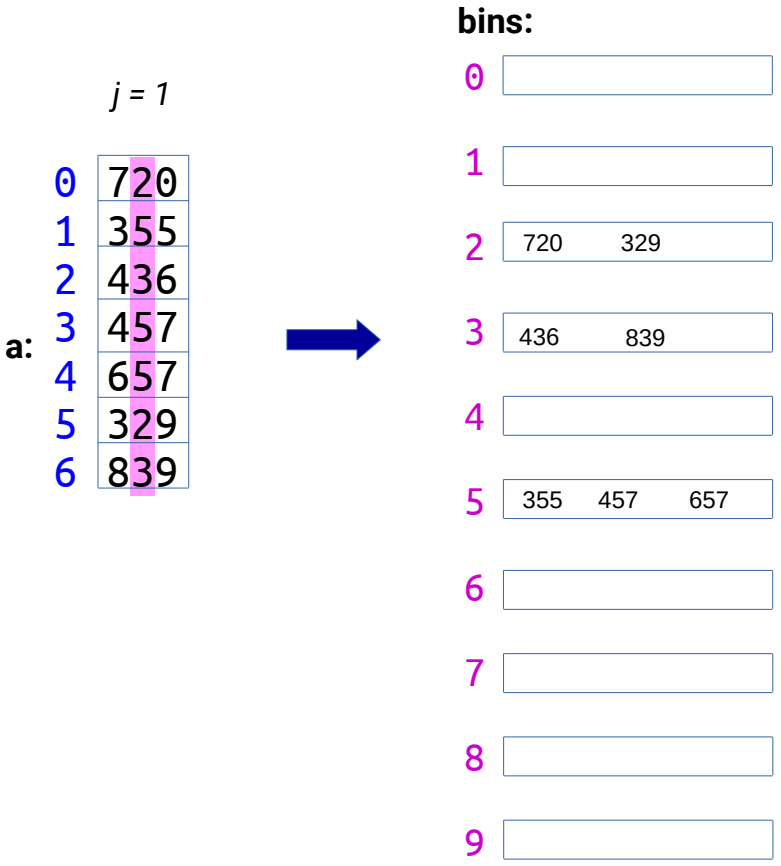
- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1*. Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins



Radix Sort

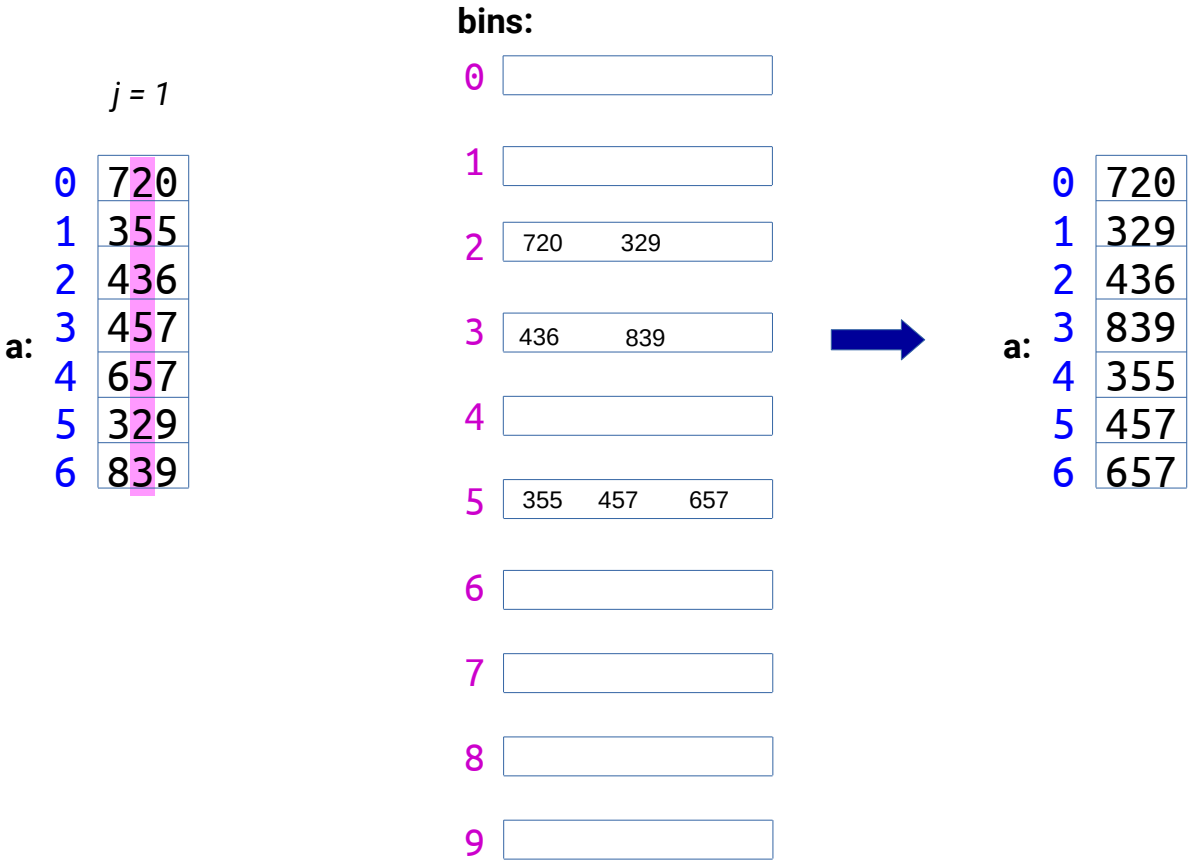
Idea:

- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1*. Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins



Radix Sort

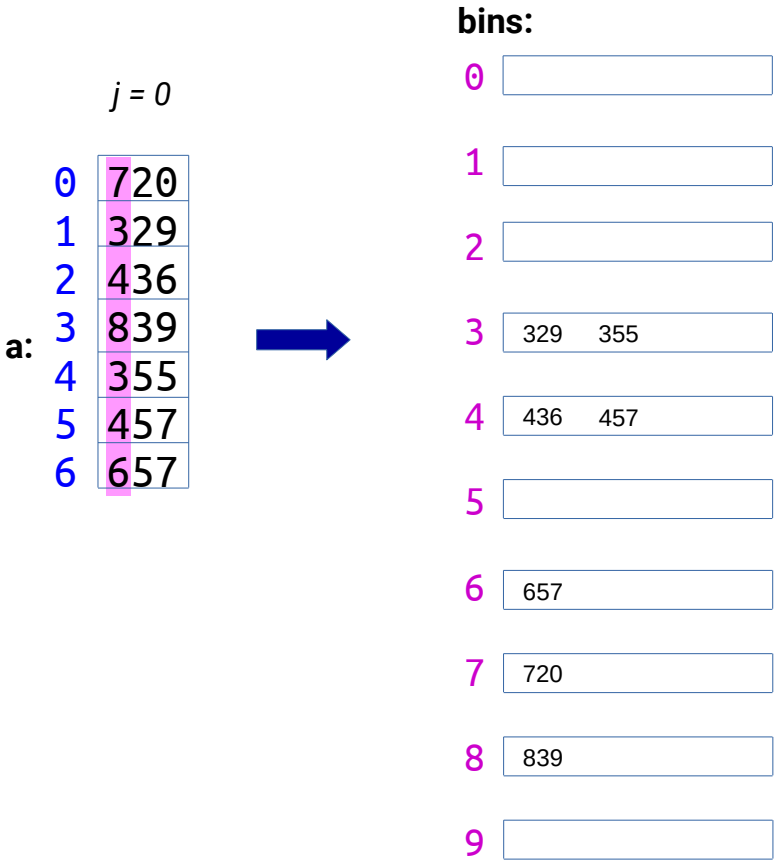
Idea:
- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1*. Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins



Radix Sort

Idea:

- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1.* Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins

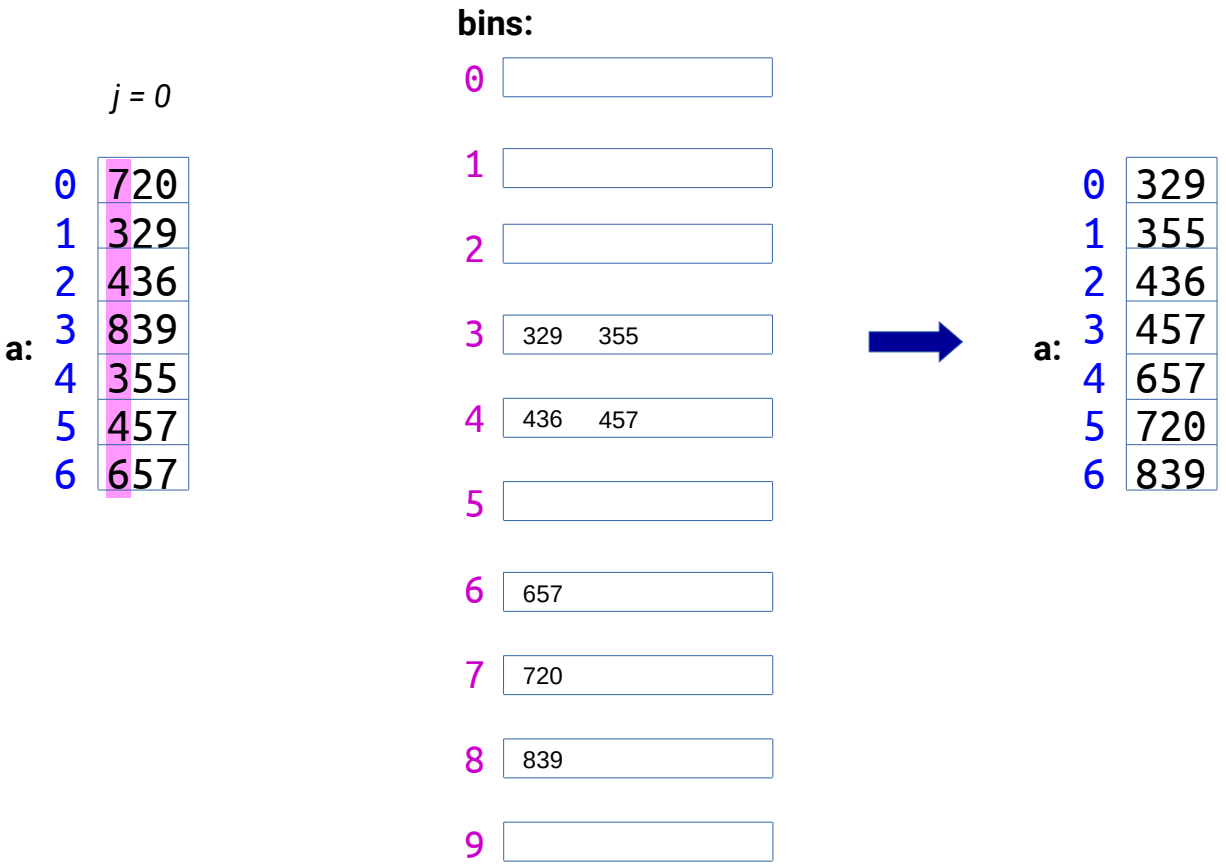


Radix Sort

O(n*d) for n numbers of d digits

Idea:

- Sort an array “a” of “n” integers, where each integer has “d” digits.
- 10 “bins” (vectors) corresponding to digits 0,…,9
- Loop, starting from the rightmost digit (“least significant”). *j* = d – 1, d – 2, …, 0
  - For i = *0,...n-1*. Insert number a[i] at the end of bin number digit(j, a[i])
  - Update array “a” by combining bins 0,…,9 (in order)
  - Clear the bins



Exercise

Implement the RadixSort algorithm to sort a vector of n strings.

input:

```
std::vector<std::string> v = {"COW", "DOG", "SEA", "RUG", "ROW", "MOB", "BOX", "TAB", "BAR", "EAR", "TAR", "DIG", "BIG", "TEA", "NOW", "FOX"};
```

output:

BAR BIG BOX COW DIG DOG EAR FOX MOB NOW ROW RUG SEA TAB TAR TEA

Notes/Hints:

- You are free to use C++ or Python! :-)
- You can assume all strings have the same number of chars!
- To ease the work, consider only uppercase letters (or only lowercase letters)
- In C++, you could create the bins as written below.

```
using Bins = std::map<char,std::vector<std::string>>;
Bins bins;
```

- How many pairs (char,std::vector) in the map (how many bins) should be initialized in the map before starting to sort the vector?
- Remember to clean the bins in your map after each iteration.
  - For C++ users, you can use bins[c].clear() to remove all elements from the bin of letter c.