

Linear Time Sorting

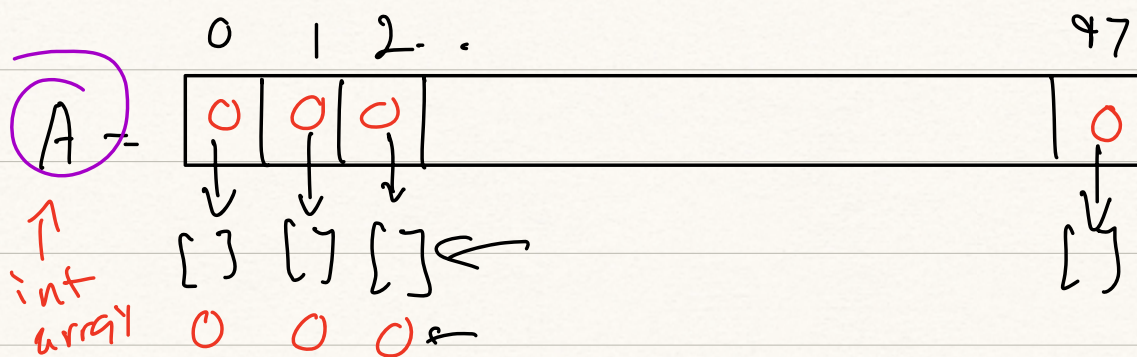
- Counting Sort

- Know: all values are between

$\{0, 1, \dots, K\}$
97

Given List L s.t. $0 \leq L[i] \leq K$
 $\forall i$, Sort L in "Linear" time

- Initialize list A of size $98 (K+1)$



Scan L :

for each i in $0, \dots, \text{size}(L)$:

$l = L[i]$

~~$A[l].append(l)$~~

$A[l]++;$

$int i = 0;$

for ($int j = 0; j < A.size(); j++$) {

$int a = A[j];$

while ($a > 0$) {

$L[i] = j;$

$i++;$ $a--;$

}

}

$L = [6, 0, 1, 5, 4, 6, 0, 4, 4]$

$A =$

0	1	2	3	4	5	6

$2 \ 1 \ 0 \ 0 \ 3 \ 1 \ 2$

↑↑↑↑↑↑↑

$L = [0, 0, 1, 4, 4, 4, 5, 6, 6]$

Time: $O(n + K)$

Space: $O(n + K)$

Radix Sort

2.100 to 0.10
+ 1.1

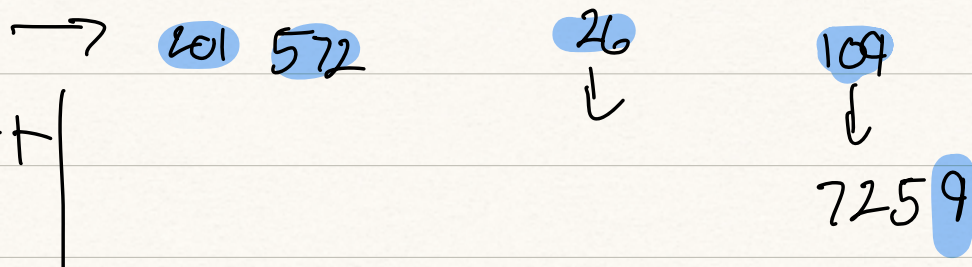
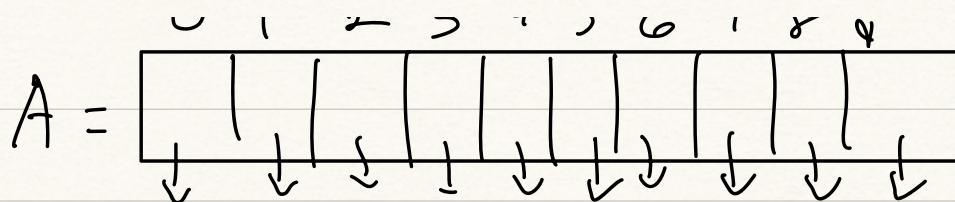
$L = [109, 26, 7259, 572, 201]$

$\begin{array}{c} 109 \\ \hline \uparrow \uparrow \end{array}, \begin{array}{c} 026 \\ \hline \uparrow \uparrow \end{array} \rightarrow \begin{array}{c} 026 \\ \hline 109 \end{array} \rightarrow \begin{array}{c} 109 \\ \hline 026 \end{array}$

$\rightarrow \begin{array}{c} 026 \\ \hline 109 \end{array} \checkmark$

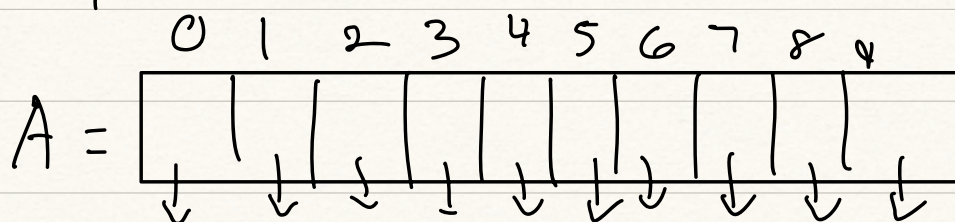
$L = [109, 26, 7259, 572, 201]$
 $\uparrow \quad \uparrow \quad \uparrow \quad \uparrow \quad \uparrow$
 $O(n)$

1 1 2 4 5 7 ~



[201, 572, 26, 109, 7259]

Digit 2

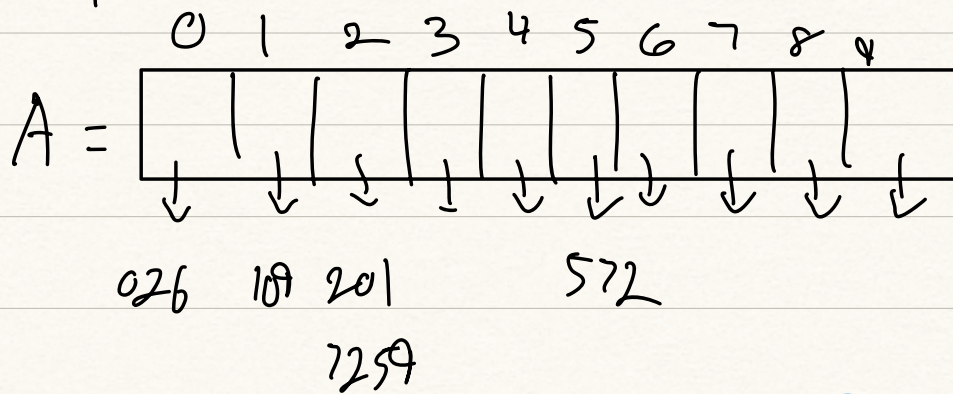


201 26 7259 572

109

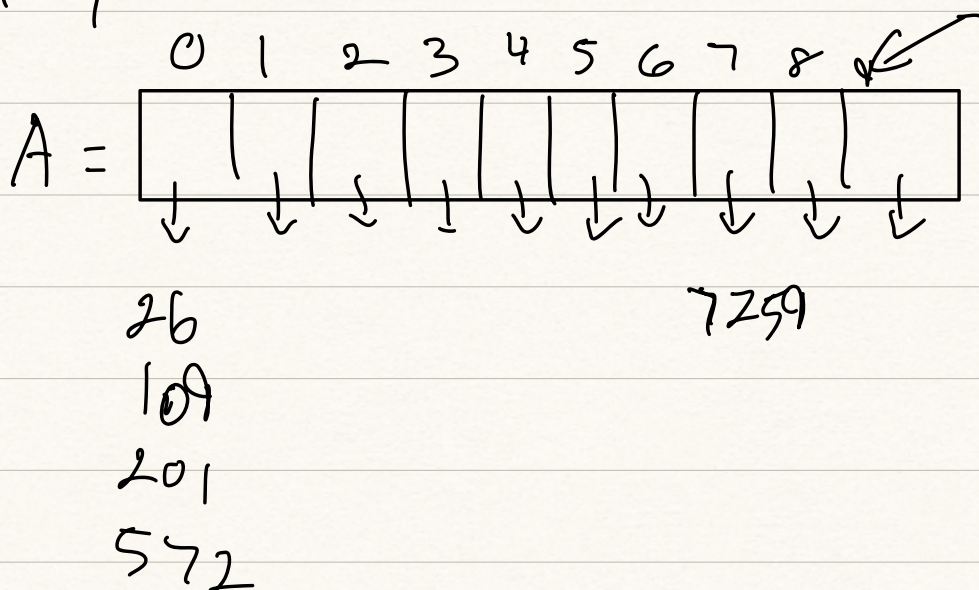
[201, 109, 26, 7259, 572]

Digit
3



[0026, 0109, 0201, 7259, 0572]

Digit
4



[26, 109, 201, 572, 7259]

... &

max
Number of digits = d

$$\Rightarrow O(d \cdot n) \text{ time}$$

$$O(n + B) \text{ space}$$

Base

$$B = 10$$

$$L = [109, 26, 7259, 572, 201]$$

$$100 \cdot 1 + 10 \cdot 0 + 1 \cdot 9$$

$$10^2 \cdot 1 + 10^1 \cdot 0 + 10^0 \cdot 9$$

$$0110101$$

$$\dots 2^2 + 2^1 + 2^0 + 2^{-1}$$

$$\underline{B=3} \quad 109 = (11001)_3$$

$$109 // 3^0 = 109$$

$$\downarrow 109 \% 3 = 1$$

$$109 // 3 = 36$$

$$36 \% 3 = 0$$

$$109 // 3^2 = 12$$

$$12 \% 3 = 0$$

$$109 // 3^3 = 4$$

$$\begin{array}{r} 36 \\ 3 \overline{) 109} \\ \underline{9} \end{array}$$

$$\underline{19}$$

$$\begin{array}{r} 12 \\ 9 \overline{) 109} \\ \underline{9} \end{array}$$

$$\begin{array}{r} 2 \\ 27 \end{array}$$

$$109 \% 3 = 1$$

$$\begin{array}{r} \times \text{ if} \\ \hline 108 \end{array}$$

$$109 // 3^4 = 1$$

Radix ($\overset{\downarrow \text{list}}{L}, \overset{\downarrow \text{base}}{B}$):

- $\text{max_val} = \max(L)$

- $d = 0$

while ($\text{max_val} // B^d > 0$) {
 $d++$;

}

/*

$$109 = 1 \cdot 3^0 + 0 \cdot 3^1 + 0 \cdot 3^2 + 1 \cdot 3^3$$

$$+ 1 \cdot 3^4 + 0 \cdot 3^5 + 0 \cdot 3^6 + 0 \cdot 3^7 \dots$$

$$3^5 > 109$$

* /

$A = \text{vector} \langle \text{vector} \langle \text{int} \rangle \rangle$
of size B

$A[0], \dots, A[B-1]$

for (int $i = 0$; $i < d$; $++i$) {

for (int $j = 0$; $j < L.size()$; $++j$) {
 $l = L[j]$

$l_digit = \text{Extract_Digit}(l, B, i)$

$A[l_digit].append(l);$

}
 $L.clear()$

for each v in A :

 while ($v.size() > 0$) {

```
    L.push(v.front());  
    v.remove_front();  
}
```

Extract_Digit(l, B, i):

return $(l // B^i) \% B$

Hybrid Sort:

Motivation:

some algorithms are
better on specific/
smaller datasets

Insertion Sorts

Very fast if

$$|L| \leq 128$$

Hybrid Merge Sort

