Welcome To My Presentation

My Presentation Topic is

Radix Sort

Presented By

Tushar Sarkar

Student ID : 18CSE35

Second Year Second Semester

Department of CSE, BSMRSTU.

Content

- Introduction
- Radix Sort Algorithm
- Working of Radix Sort
- Pseudocode
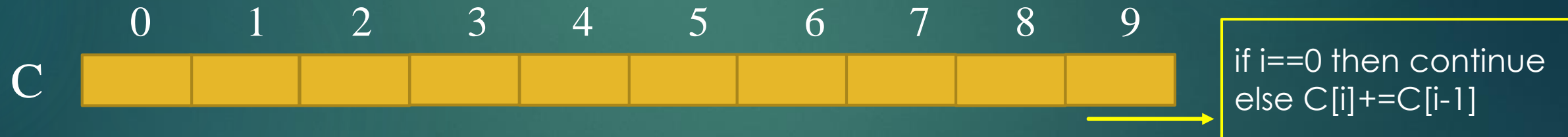- Time Complexity
- Conclusion

Introduction

▶ Radix sort is *a sorting algorithms* that sorts the elements by first grouping the individual digits of the same **place value**.

▶ Then, sort the elements according to their increasing/decreasing order.

▶ We have seen many sorting algorithms but this one is different method.

▶ It is interesting because it requires the absolute minimum amount of apace and the minimum amount of data movement and most amazing off all, it does no comparisons.

Radix Sort Algorithm

▶ Find the largest element in the array, i.e. max. Let X be the number of digits in max . X is calculated because we have to go through all the significant places of all elements.

▶ Now, go through each significant place one by one. Use any stable sorting technique to sort the digits at each significant place. We have used counting sort for this

▶ First, we will sort elements based on the value of the unit place. Then, we will sort elements based on the value of the tenth place. This process goes on until the last significant place.

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

Main Array

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

First , C[10]=0
Then i=0 to 9
    C[(A[i]/pos)%10]++

	0	1	2	3	4	5	6	7	8	9
C										

if i==0 then continue
else C[i]+=C[i-1]

	0	1	2	3	4	5
Output						

Output Array
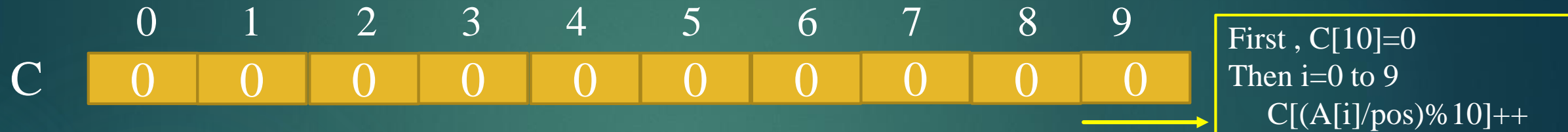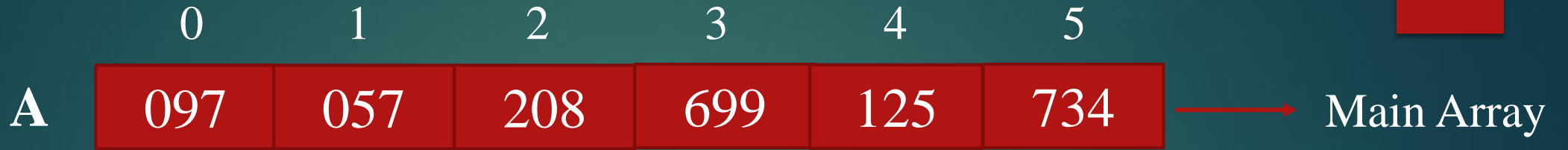
Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=0 & pos=1
C[(097/1)%10]++
=C[7]++
Then , C[7]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=1 & pos=1
C[(057/1)%10]++
=C[7]++
Then , C[7]=2

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=2 & pos=1
C[(208/1)%10]++
=C[8]++
Then , C[8]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=3 & pos=1
C[(699/1)%10]++
=C[9]++
Then , C[9]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=4 & pos=1
C[(125/1)%10]++
=C[5]++
Then , C[5]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=5 & pos=1
C[(734/1)%10]++
=C[4]++
Then , C[4]=1
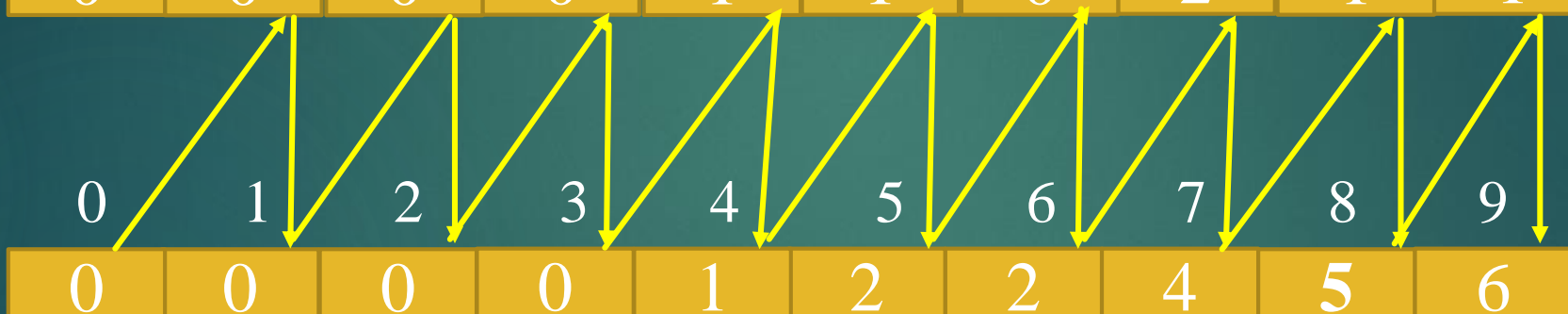
	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

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

If I == 0 then continue
Else c[i]+=C[i-1]

	0	1	2	3	4	5
Output						

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=5 & pos=1
C[(734/1)%10]--
=C[4]-1,C[4]=0
Output[0]=734

	0	1	2	3	4	5
Output	734					

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=3 & pos=1
C[(699/1)%10]--
=C[9]-1,C[9]=5
Output[5]=699

	0	1	2	3	4	5
Output	734	125				699

Working of Radix Sort(Unit Place)

A

	0	1	2	3	4	5
A	097	057	208	699	125	734

C

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

If i=2 & pos=1
C[(208/1)%10]--
=C[8]-1,C[8]=4
Output[4]=208

Output

	0	1	2	3	4	5
Output	734	125			208	699

Working of Radix Sort(Unit Place)

A

0	1	2	3	4	5
097	057	208	699	125	734

C

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

If i=1 & pos=1
C[(057/1)%10]--
=C[7]-1,C[7]=3
Output[3]=057

Output

0	1	2	3	4	5
734	125		057	208	699

Working of Radix Sort(Unit Place)

	0	1	2	3	4	5
A	097	057	208	699	125	734

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

If i=0 & pos=1
C[(097/1)%10]--
=C[7]-1,C[7]=2
Output[2]=097

	0	1	2	3	4	5
Output	734	125	097	057	208	699

Convert to Main Array

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

→ Main Array

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

First , C[10]=0
Then i=0 to 9
    C[(A[i]/pos)%10]++

	0	1	2	3	4	5	6	7	8	9
C										

if i==0 then continue
else C[i]+=C[i-1]

0	1	2	3	4	5

Output

→ Output Array
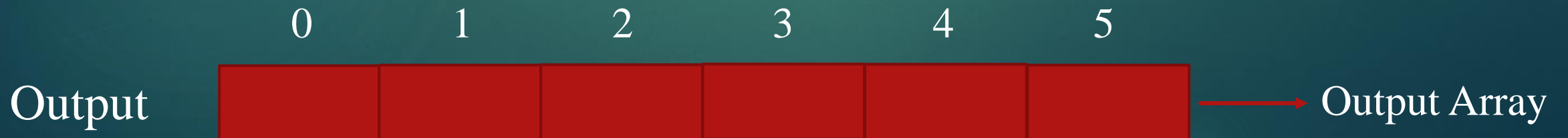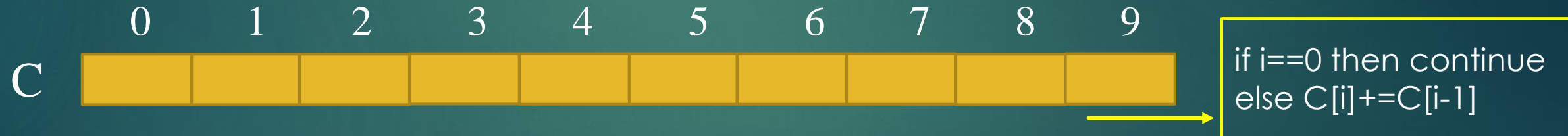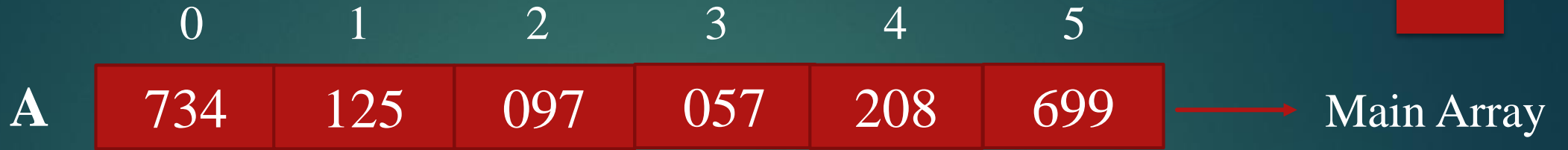
Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=0 & pos=10
C[(734/10)%10]++
=C[3]++
Then , C[3]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=1 & pos=10
C[(125/10)%10]++
=C[2]++
Then , C[2]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=2 & pos=10
C[(097/10)%10]++
=C[9]++
Then , C[9]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=3 & pos=10
C[(057/10)%10]++
=C[5]++
Then , C[5]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=4 & pos=10
C[(208/10)%10]++
=C[0]++
Then , C[0]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=5 & pos=10
C[(699/10)%10]++
=C[9]++
Then , C[9]=2
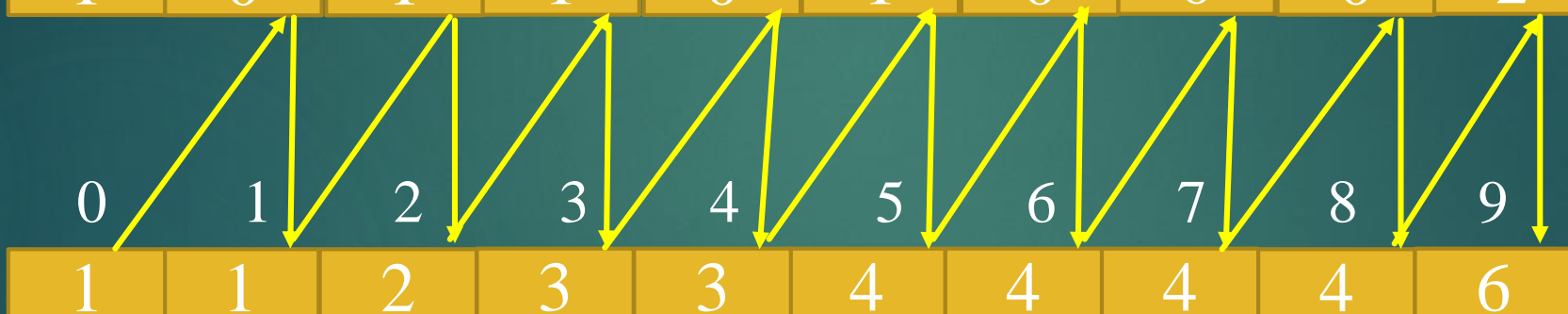
	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

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

If I == 0 then continue
Else c[i]+=C[i-1]

	0	1	2	3	4	5
Output						

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=5 & pos=10
C[(699/10)%10]--
=C[9]-1,C[9]=5
Output[5]=699

	0	1	2	3	4	5
Output						699

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

	0	1	2	3	4	5	6	7	8	9
C	0	1	2	3	3	4	4	4	4	5
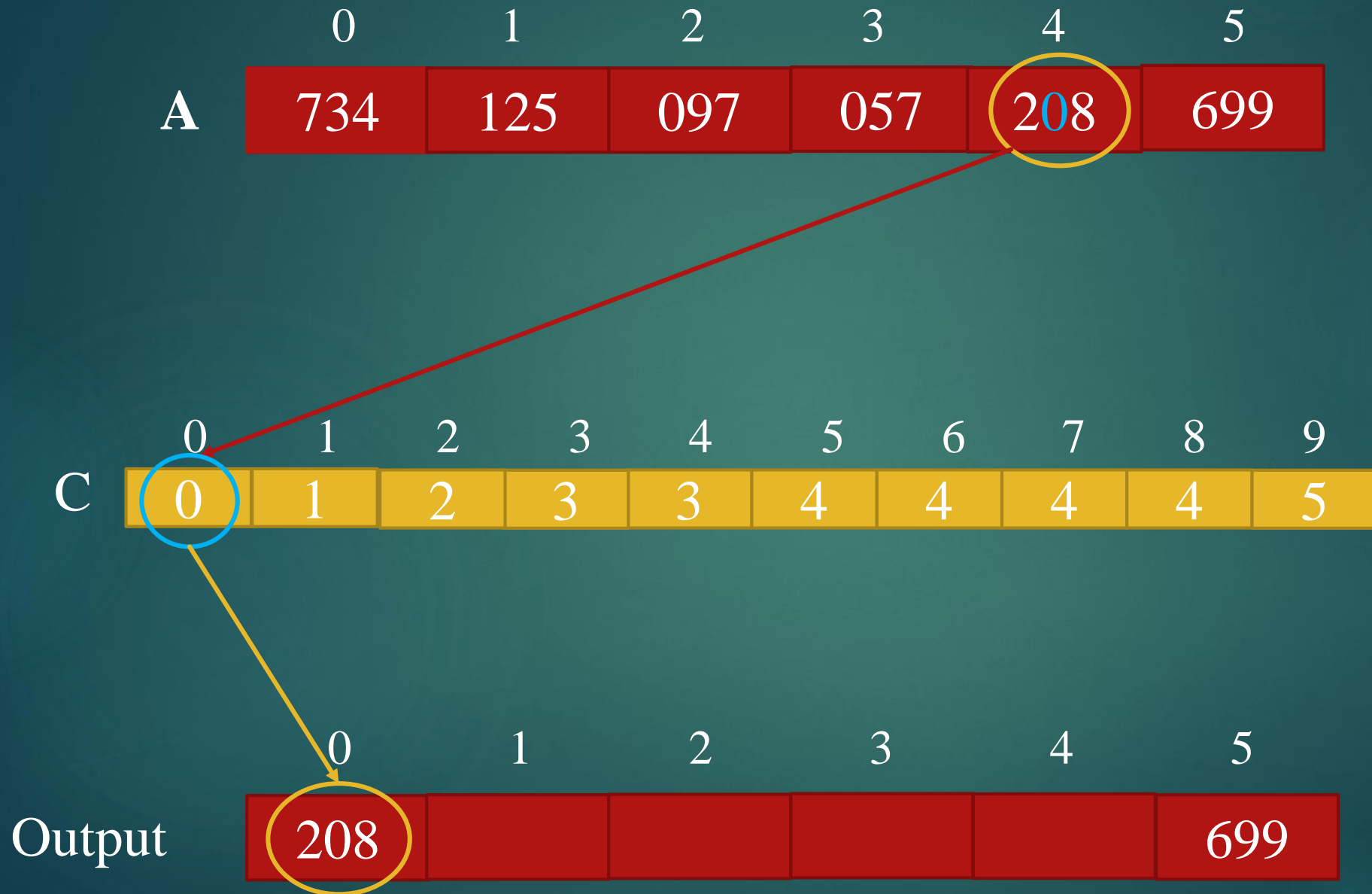
If i=4 & pos=10
C[(208/10)%10]--
=C[0]-1,C[0]=0
Output[0]=208

	0	1	2	3	4	5
Output	208					699

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=3 & pos=10
C[(057/10)%10]--
=C[5]-1,C[5]=3
Output[3]=057

	0	1	2	3	4	5
Output	208			057		699

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=2 & pos=10
C[(097/10)%10]--
=C[9]-1,C[9]=4
Output[4]=097

	0	1	2	3	4	5
Output	208			057	097	699

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=1 & pos=10
C[(125/10)%10]--
=C[2]-1,C[2]=1
Output[1]=125

	0	1	2	3	4	5
Output	208	125		057	097	699

Working of Radix Sort(Tens Place)

	0	1	2	3	4	5
A	734	125	097	057	208	699

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

If i=0 & pos=10
C[(734/10)%10]--
=C[3]-1,C[3]=2
Output[2]=734

	0	1	2	3	4	5
Output	208	125	734	057	097	699

Convert to Main Array

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

Main Array →

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

First , C[10]=0
Then i=0 to 9
    C[(A[i]/pos)%10]++

	0	1	2	3	4	5	6	7	8	9
C										

if i==0 then continue
else C[i]+=C[i-1]

	0	1	2	3	4	5
Output						

Output Array →

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=0 & pos=100
C[(208/100)%10]++
=C[2]++
Then , C[2]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=1 & pos=100
C[(125/100)%10]++
=C[1]++
Then , C[1]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=2 & pos=100
C[(734/100)%10]++
=C[7]++
Then , C[7]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=3 & pos=100
C[(057/100)%10]++
=C[0]++
Then , C[0]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=4 & pos=100
C[(097/100)%10]++
=C[0]++
Then , C[0]=2

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=5 & pos=100
C[(699/100)%10]++
=C[6]++
Then , C[6]=1

	0	1	2	3	4	5	6	7	8	9
C										

	0	1	2	3	4	5
Output						

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

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

If I == 0 then continue
Else c[i]+=C[i-1]
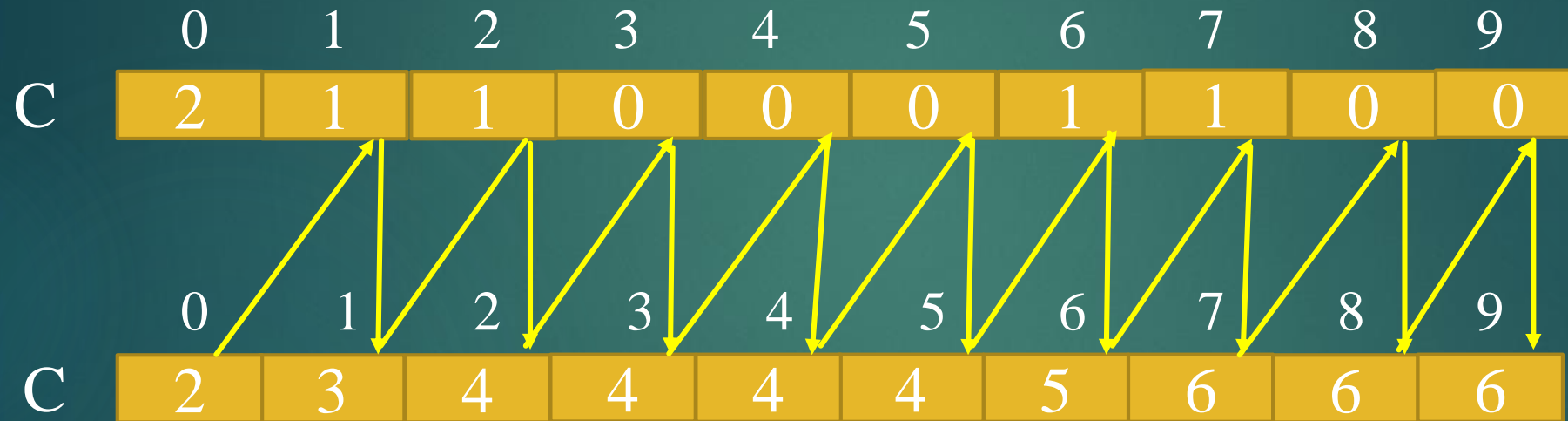
	0	1	2	3	4	5
Output						

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=5 & pos=100
C[(699/100)%10]--
=C[6]-1,C[6]=4
Output[4]=699

	0	1	2	3	4	5
Output					699	

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=4 & pos=100
C[(097/100)%10]--
=C[0]-1,C[0]=1
Output[1]=097

	0	1	2	3	4	5
Output		097			699	

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=3 & pos=100
C[(057/100)%10]--
=C[0]-1,C[0]=0
Output[0]=057

	0	1	2	3	4	5
Output	057	097			699	

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=2 & pos=100
C[(734/100)%10]--
=C[7]-1,C[7]=5
Output[5]=734

	0	1	2	3	4	5
Output	057	097			699	734

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

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

If i=1 & pos=100
C[(125/100)%10]--
=C[1]-1,C[1]=2
Output[2]=125

	0	1	2	3	4	5
Output	057	097	125		699	734

Working of Radix Sort(Hundreds Place)

	0	1	2	3	4	5
A	208	125	734	057	097	699

If i=0 & pos=100
C[(208/100)%10]--
=C[2]-1,C[2]=3
Output[3]=208

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

	0	1	2	3	4	5
Output	057	097	125	208	699	734

Convert to Main Array

Working of Radix Sort

	0	1	2	3	4	5
A	097	057	208	699	125	734

Given Array & Unsorted

Output

	0	1	2	3	4	5
A	057	097	125	208	699	734

Sorted Array

Pseudocode

```cpp
88  void counting_sort(ll A[], ll n, ll pos)
89  {
90      ll ctlen = 10, i;
91      ll count[ctlen], Output[n];
92
93      for (i = 0; i < ctlen; i++)count[i] = 0;
94      for (i = 0; i < n; i++)count[(A[i] / pos) % 10]++;
95      for (i = 1; i < ctlen; i++)count[i] += count[i - 1];
96      for (i = n - 1; i >= 0; i--)Output[--count[(A[i] / pos) % 10]] = A[i];
97      for (i = 0; i < n; i++)A[i] = Output[i];
98  }
99  void radix_sort(ll A[], ll n)
100 {
101     ll i, mx = 0, pos;
102     Forn(i, n)mx = max(mx, A[i]);
103     for (pos = 1; mx / pos > 0; pos *= 10)counting_sort(A, n, pos);
104 }
105
106 int main()
107 {
108     ll n, i;
109     cout << "Enter the value of n : ";
110     cin >> n;
111     ll A[n];
112     cout << "Enter the element of A : ";
113     Forn(i, n)cin >> A[i];
114     radix_sort(A, n);
115     cout << "Output : ";
116     Forn(i, n)cout << A[i] << " ";
117     cout << endl;
118
119     biday;
120 }
121
```

```
C:\WINDOWS\system32\cmd.exe - pause

Enter the value of n : 6
Enter the element of A : 97 57 208 699 125 734
Output : 57 97 125 208 699 734
Press any key to continue . . .
```

Time Complexity

❖ The time complexity of radix sort is given by the formula,

$$T(n) = O(d*(n+b))$$

Where,

➢ d is the number of digits in the given listn is the number of elements in the list

➢ n is the number of elements in the list

➢ b is the base or bucket size used, which is normally base 10 for decimal representation.

Thank You