

Assignment Project Exam Help

Linear Sorting

<https://tutorcs.com>

Dr. Timothy Kimber

WeChat: [cstutorcs](#)

March 2018

Recalling Comparison Sorts

Assignment Project Exam Help

The running time of these comparison sort algorithms

- Mergesort
- Heapsort
- Quicksort (expected)

<https://tutorcs.com>

are all $O(N \log N)$.

- Not possible for a comparison sort algorithm to do better

WeChat: cstutorcs

However, there are sorting methods that achieve $O(N)$ performance.

Counting Sort

The **Counting Sort** algorithm sorts integers from a **known range**

- The key operation is to count the occurrences of all values

Counting Sort(Input: $A = [A_1, \dots, A_N]$, k)

- For $i = 0$ to k
 - $C[i] = 0$ \leftarrow one entry per value in the range
- For $j = 1$ to N
 - $C[A[j]] = C[A[j]] + 1$ \leftarrow count how many $A[j]$ there are
- For $i = 1$ to k
 - $C[i] = C[i] + C[i-1]$ \leftarrow how many less than or equal to i
- For $j = N$ to 1
 - $B[C[A[j]]] = A[j]$
 - $C[A[j]] = C[A[j]] - 1$
- Return B

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

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

B								
-----	--	--	--	--	--	--	--	--

	0	1	2	3	4	5
C						

Counting Sort

Assignment Project Exam Help

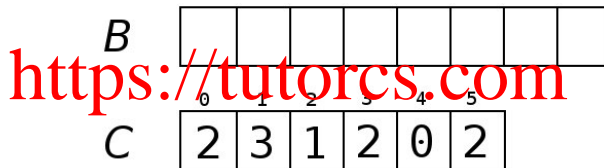


WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

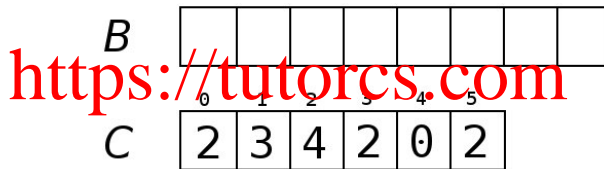


WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

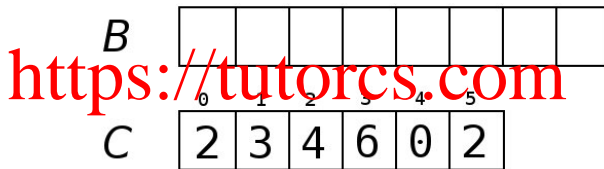


WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help



WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help



WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

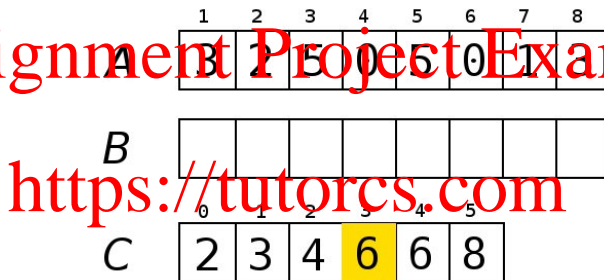


WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help



WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

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

B						3		
	0	1	2	3	4	5		

C	2	3	4	6	6	8
---	---	---	---	---	---	---

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

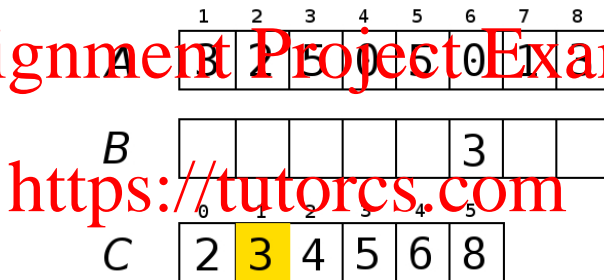
	1	2	3	4	5	6	7	8
A	3	1	5	0	6	0	1	

B						3		
	0	1	2	3	4	5		

C	2	3	4	5	6	8
---	---	---	---	---	---	---

Counting Sort

Assignment Project Exam Help



WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

B

		1			3		
0	1	2	3	4	5		

C

2	3	4	5	6	8
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

B

		1			3		
0	1	2	3	4	5		

C

2	2	4	5	6	8
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

B

		1			3		
0	1	2	3	4	5		

C

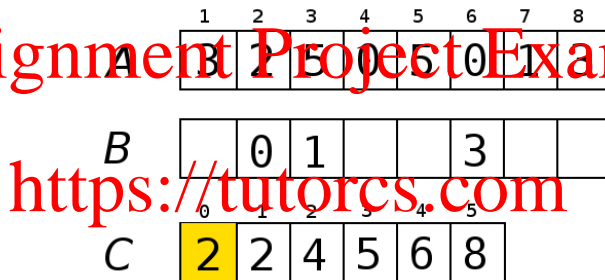
2	2	4	5	6	8
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help



WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

Counting Sort

Assignment Project Exam Help

B

	0	1			3		
0	1	2	3	4	5		

C

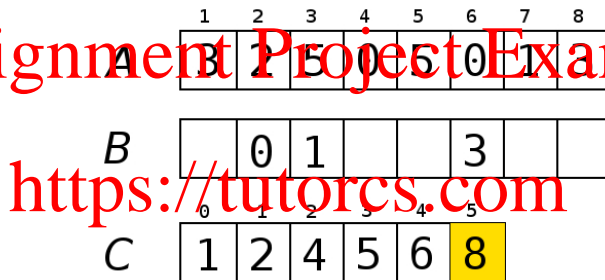
1	2	4	5	6	8
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help



WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

Counting Sort

Assignment Project Exam Help

B

	0	1			3		5
0	1	2	3	4	5		

C

1	2	4	5	6	8
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

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

B		0	1			3		5
	0	1	2	3	4	5		

C	1	2	4	5	6	7		
---	---	---	---	---	---	---	--	--

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

Counting Sort

Assignment Project Exam Help

B

	0	1			3		5
0	1	2	3	4	5		

C

1	2	4	5	6	7
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

B

0	0	1			3		5
0	1	2	3	4	5		

C

1	2	4	5	6	7
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

B

0	0	1			3		5
0	1	2	3	4	5		

C

0	2	4	5	6	7
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

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

B	0	0	1			3		5
	0	1	2	3	4	5		

C	0	2	4	5	6	7
---	---	---	---	---	---	---

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

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

B	0	0	1			3	5	5
	0	1	2	3	4	5		

C	0	2	4	5	6	7
-----	---	---	---	---	---	---

Counting Sort

Assignment Project Exam Help

	1	2	3	4	5	6	7	8
A	3	1	5	0	6	0	1	0
B	0	0	1			3	5	5
	0	1	2	3	4	5		
C	0	2	4	5	6	6		

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

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

B	0	0	1			3	5	5
	0	1	2	3	4	5		

C	0	2	4	5	6	6
-----	---	---	---	---	---	---

Counting Sort

Assignment Project Exam Help

B

0	0	1	2		3	5	5
0	1	2	3	4	5		

C

0	2	4	5	6	6
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

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

B	0	0	1	2		3	5	5
	0	1	2	3	4	5		

C	0	2	3	5	6	6
-----	---	---	---	---	---	---

Counting Sort

Assignment Project Exam Help

B

0	0	1	2		3	5	5
0	1	2	3	4	5		

C

0	2	3	5	6	6
---	---	---	---	---	---

WeChat: cstutorcs

- Counts of each value are saved into *C*
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of *A* to correct positions in *B* using *C*

Counting Sort

Assignment Project Exam Help

<https://tutorcs.com>

	1	2	3	4	5	6	7	8
A	3	1	5	0	6	0	1	
B	0	0	1	2	3	3	5	5
	0	1	2	3	4	5		
C	0	2	3	5	6	6		

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

Counting Sort

Assignment Project Exam Help

	1	2	3	4	5	6	7	8
A	3	1	5	0	6	0	1	
B	0	0	1	2	3	3	5	5
	0	1	2	3	4	5		
C	0	2	3	4	6	6		

WeChat: cstutorcs

- Counts of each value are saved into C
- Next the counts are accumulated
- Now $C[i]$ holds number of values $\leq i$
- Finally copy contents of A to correct positions in B using C

Counting Sort Time

Counting sort makes two passes through the input and two passes through the count table C

- So the time taken is ...

Counting Sort(Input: $A = [A_1, \dots, A_N]$, k)

- For $i = 0$ to k
 - $C[i] = 0$ \leftarrow one entry per value in the range
- For $j = 1$ to N
 - $C[A[j]] = C[A[j]] + 1$ \leftarrow count how many $A[j]$ there are
- For $i = 1$ to k
 - $C[i] = C[i] + C[i - 1]$ \leftarrow how many less than or equal to i
- For $j = N$ to 1
 - $B[C[A[j]]] = A[j]$
 - $C[A[j]] = C[A[j]] - 1$
- Return B

Properties

Assignment Project Exam Help

Counting Sort runs in $\Theta(N + k)$ time.

Question

Under what circumstances does this become $O(N)$ time?

Counting Sort is also **stable**

- 'Different' 3s stay in the same order
- Can be important when the values are linked to other data
- This property is used by the next algorithm

Radix Sort

Radix Sort is used to sort a set of d -digit values

Assignment Project Exam Help

535 089
 158 134
 189 158
 134 189
 840 535
 558 558
 089 840

<https://tutorcs.com>

WeChat: cstutorcs

- It makes d passes through the data
- Each pass sorts on the i th digit only

Radix Sort

Radix Sort is used to sort a set of d -digit values

Assignment Project Exam Help

<https://tutorcs.com>

WeChat: cstutorcs

- Counter-intuitively, the first sort is on the **least significant digit**
- It allows counting sort to be used per digit, over a much smaller range
- e.g. For decimal numbers there are 10 values to sort on

Radix Sort

Radix Sort is used to sort a set of d -digit values

Assignment Project Exam Help

840

134

535

158

558

189

089

<https://tutorcs.com>

WeChat: cstutorcs

- Counter-intuitively, the first sort is on the **least significant digit**
- It allows counting sort to be used per digit, over a much smaller range
- e.g. For decimal numbers there are 10 values to sort on

Radix Sort

Radix Sort is used to sort a set of d -digit values

Assignment Project Exam Help

840

134

535

<https://tutorcs.com>

158

558

189

WeChat: cstutorcs

089

- Counter-intuitively, the first sort is on the **least significant digit**
- It allows counting sort to be used per digit, over a much smaller range
- e.g. For decimal numbers there are 10 values to sort on

Radix Sort

Radix Sort is used to sort a set of d -digit values

Assignment Project Exam Help

134

535

840

<https://tutorcs.com>

158

558

189

WeChat: cstutorcs

089

- Counter-intuitively, the first sort is on the **least significant digit**
- It allows counting sort to be used per digit, over a much smaller range
- e.g. For decimal numbers there are 10 values to sort on

Radix Sort

Radix Sort is used to sort a set of d -digit values

Assignment Project Exam Help

134

535

840

<https://tutorcs.com>

158

558

189

089

WeChat: cstutorcs

- Counter-intuitively, the first sort is on the **least significant digit**
- It allows counting sort to be used per digit, over a much smaller range
- e.g. For decimal numbers there are 10 values to sort on

Radix Sort

Radix Sort is used to sort a set of d -digit values

Assignment Project Exam Help

089

134

158

<https://tutorcs.com>

189

535

558

WeChat: cstutorcs

840

- Counter-intuitively, the first sort is on the **least significant digit**
- It allows counting sort to be used per digit, over a much smaller range
- e.g. For decimal numbers there are 10 values to sort on

Radix Sort

Assignment Project Exam Help

The algorithm is simple to state

Radix Sort(Input: $A = [A_1, \dots, A_N], d$)

- For $i = 0$ to d
 - Use a stable sort to sort A on digit i
- Counting Sort can implement the stable sort efficiently

The Radix

Assignment Project Exam Help

Radix Sort (Input: $A = [A_1, \dots, A_N], d$)

- For $i = 0$ to d
 - Use a stable sort to sort A on digit i

<https://tutorcs.com>

Discussion

You are sorting N numbers with Radix sort. You can *choose* what **base** the numbers will be represented in within the sort procedure.

- What base would you choose?
- Why?

WeChat: cstutorcs

The Radix

Assignment Project Exam Help

Assuming we have N numbers

- Expressed in base B
- Each with up to d digits

Radix sort takes $d(N + B)$ time.

- Base B has values in the range 0 to $(B - 1)$
- So, there are B distinct values to count

A base that is $O(N)$, e.g. base N , will limit the number of digits compared to some smaller base, while not dominating the time for each pass.

Binary

Binary representation allows you to pick any power of 2 as a base very cheaply. Assuming we have N numbers

- Each number has b bits
- Split the number into digits each comprising r bits

Radix Sort runs in $\Theta((b/r)(N + 2^r))$ time (if the stable sort takes $\Theta(N + k)$ time to sort values in the range $0 \dots k$).

- Each number has b/r digits
- Choose $r \sim \log_2(N)$ gives $\sim N$ values per digit

Under the assumption that $b = O(\log_2 N)$ the running time of Radix Sort is $\Theta(N)$. In practice, constant factors may mean that Quicksort is faster.