Insertion and Radix Sort

Jeasmin(2003017)
Jerin(2003032)
Neha(2003034)
Srishty(2003042)
Ania(2003048)
Sara(2003058)

Group 8

January 23, 2024

Presentation outline

Insertion Sort

Insertion sort[**isDF**] is a sorting algorithm that places an unsorted element at its suitable place in each iteration.

Example:

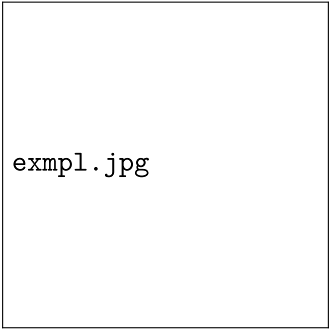


Fig 1[**isDF**]: Sorting an array using insertion sort

How does it work?

is1.jpg

[isDF]

Step 1

- Initial array

How does it work?

is1.jpg

[isDF]

Step 1

- Initial array

How does it work?

is1.jpg

[isDF]

Step 1

- Initial array

How does it work?

is1.jpg

[isDF]

Step 1

- Initial array

How does it work?

Step 2

is3.jpg

[isDF]

- key=1

How does it work?

Step 3

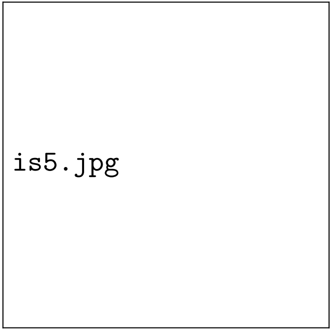
is4.jpg

[isDF]

- key=4

How does it work?

Step 4



- key=3

Psudocode

ispsudo.jpg

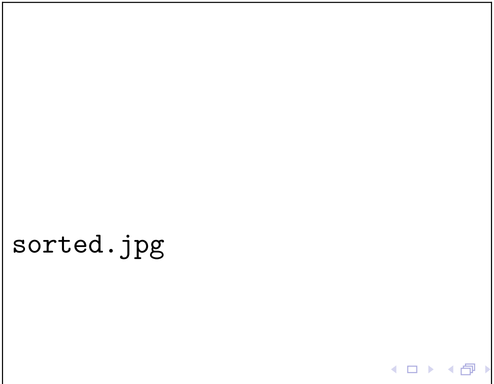
Advantages/Disadvantages

Advantages[ISad]	**Disadvantages[ISad]**
The main advantage of the insertion sort is its simplicity.	The disadvantage of the insertion sort is that it does not perform as well as other, better sorting algorithms
It also exhibits a good performance when dealing with a small list.	With n-squared steps required for every n element to be sorted, the insertion sort does not deal well with a huge list.
The insertion sort is an in-place sorting algorithm so the space requirement is minimal.	The insertion sort is particularly useful only when sorting a list of few items.

Radix Sort

Radix sort[**radDef**] is a non-comparative sorting algorithm. Radix sort was developed to sort large integers. As an integer is treated as a string of digits so we can also call it a string sorting algorithm.

Example:



sorted.jpg

How does it work?

Radix-Sort-Array.png

- Initial array

How does it work?

Radix-Sort--1.png

- focus on least significant bit(LSB) and sort array according to LSB

How does it work?

Radix-Sort--2.png

- sort according to 10th possition

How does it work?

Radix-Sort--3.png

- sort according to 100th possition

How does it work?

Radix-Sort--4.png

- sorted array

Psudocode

pseudocode.png

Advantages/Disadvantages

Advantages[rad3]	**Disadvantages[rad3]**
Radix sort is fast when the keys are short i.e. when the range of the array elements is less.	Since it depends on digits or letters, Radix Sort is much less flexible than other sorts.
It is a stable sorting algorithm, meaning that elements with same key value maintain their relative order in sorted output.	It requires a significant amount of memory to hold the count of the number of times each digit value appears.
It has a linear time complexity,which makes it faster than comparison-based sorting algorithms	It is not efficient for small data sets or data sets with a small number of unique keys.

Time & Space Complexity

Time Complexity:

Insertion Sort has a time complexity of $O(n^2)$ for average case and worst case and $O(n)$ for best case, where n is the size of array.

Radix Sort has a time complexity of $O(nd)$, where n is the size of array and d is number of digits.

Space Complexity:

Insertion Sort has a space complexity of $O(1)$.

Radix Sort has a space complexity of $O(n + d)$.

References

thankyou.png