1 Comparison Based Sorting Lowerbound

Assume you have a function called 3-sort that takes as input an array of size 3 of comparable objects and returns them in sorted order. You want to sort an array of size \( n \geq 3 \) objects only using calls to 3-sort. How many function calls of 3-sort are required to sort an array of size \( n \)?

- \( \Omega(\log(n)) \)
- \( \Omega(\sqrt{n}) \)
- \( \Omega(n) \)
- \( \Omega(n\log n) \)

\( \Omega(\log(n)) \) is correct.

Is it possible to implement the following algorithms or data structures in real life?

A data structure that you can insert comparable objects in \( O(1) \) operations, remove them in \( O(1) \) operations, and make function calls to a function that returns the smallest object in the data structure in \( O(1) \)?

- Possible
- Impossible

Possible is correct.

An algorithm that runs in \( O(n) \) and takes and array \( A \) of size \( n \) of comparable objects, \( i \), and \( j \) (\( i < j \)) as inputs and returns all the elements that are greater than the \( i \)th element in \( A \) and less than the \( j \)th element in \( A \)?

- Possible
- Impossible

Possible is correct.

Which algorithm would be a good choice to use for the last part?

- k-select
- Radix Sort
- QuickSort
- MergeSort

QuickSort is correct.

2 Radix Sort and Counting Sort

Assume you have an array \( A \) of size \( n \) with positive integer element with all elements in range of \([1, n^3]\).

What is the runtime of Counting sort run on \( A \)?

- \( \Theta(n) \)
- \( \Theta(n\log n) \)
- \( \Theta(n^2) \)
- \( \Theta(n^3) \)

\( \Theta(n) \) is correct.

What is the runtime of Radix sort, base 10, run on \( A \)?

- \( \Theta(n) \)
- \( \Theta(n\log n) \)
- \( \Theta(n^2) \)
- \( \Theta(n^3) \)

\( \Theta(n) \) is correct.

Which base for Radix sort will result in the fastest Radix sort algorithm to sort \( A \) with?

- 2
- 10
- \( n \)
- \( n^2 \)

\( n \) is correct.