

Data Structures and Algorithms – CS2402

Exercises on Sorting

List of sorting algorithms to be presented:

1. bubble sort *presented on Wednesday by Julio*
2. insertion sort *presented on Wednesday by Vladimir*
3. selection sort *to be presented by Christian, prepared with her group*
4. merge sort *to be presented by Luis Garcia, prepared with his group*
5. heap sort *to be presented by Michael Harney, prepared with his group*

* in class we will also review quick sort (of course :-)).

Exercise 1 *As is presented in all algorithms we've reviewed so far, data are stored in an array and this array is the data structure to be sorted. What are the pros and cons of using an array?*

- *for sorting;*
- *to store the data to be sorted;*
- *for future use of the sorted data.*

Discuss these points keeping complexity of operations in mind.

Exercise 2 *Suppose you have all numbers to be sorted stored in an array. Which of the above-mentioned algorithms would you use? Discuss your answer depending on the parameters of your problem.*

Exercise 3 *Suppose you read from an input stream a series of numbers to be sorted. Would you use one of the above-mentioned algorithms, or would you proceed differently? In either case, justify your answer by showing that your method would be the most efficient (cf. time complexity).*

Exercise 4 *Up to (roughly) which size of the input data is it better to use a n^2 algorithm vs. a $n \log(n)$ algorithm?*

Exercise 5 *Are all n^2 sorting algorithms equivalent in terms of performance?*

Exercise 6 *Earlier in the semester, we have studied BST, AVL, heaps, and we've seen that these structures are also convenient for sorting, searching, etc.*

1. *Compare handling these sorted structures to sorting arrays.*
2. *When is it better to use arrays? trees?*

Exercise 7 *What about hash tables? They are meant for easy access of data. Isn't it what we expect from sorting data? So what is the point of sorting? and not hashing?*