Questions

Data Structures & Algorithms Labs
Week 3: Sorting

These exercises will allow you to test your knowledge of the lecture material from week 2 (sorting).

1 Programming Exercises

Once you have solved each of these exercises, try to figure out the (average/worst case) time and space complexity of your solution, and whether more efficient solutions exist.

► **1.1.** Write a function that takes two sorted arrays, both in non-decreasing order, and outputs a single sorted array containing all the elements of both input arrays. Can you do it in linear time?

Example: $[1, 3, 4, 5, 6, 8, 9], [1, 2, 3, 4] \to [1, 1, 2, 3, 3, 4, 4, 5, 6, 8, 9]$ ◄

► **1.2.** Write a stable implementation of the quicksort algorithm. Stable means that all pairs of objects with equal sorting keys will appear in the same relative order in the output as they do in the input. In other words, if two elements are equivalent then they won't be swapped with each other. ◄

► **1.3.** Write a function that takes an array and returns an array with only the unique elements from the input array (duplicates removed).

Example: $[1, 9, 3, 1, 1, 5, 6, 9, 5] \to [1, 3, 5, 6, 9]$ ◄

► **1.4.** Write a function that takes two input arrays, `arr1` and `arr2`, of lengths `m` and `n` respectively, and outputs `true` if every element of `arr1` is in `arr2`, and `false` otherwise.

Example 1: $[2, 5, 3], [3, 2, 1, 4, 5] \to$ `true`

Example 2: $[1, 6], [3, 2, 1, 4, 5] \to$ `false` ◄

► **1.5.** Write a function that takes an `m*n` matrix of integers from `0-9`, and returns the same matrix with its rows sorted lexicographically in increasing order, as if each row were a single number.

Example: $\begin{bmatrix} 1 & 7 & 8 & 2 \\ 1 & 3 & 3 & 3 \\ 9 & 1 & 2 & 3 \\ 5 & 6 & 2 & 9 \\ 1 & 7 & 1 & 9 \end{bmatrix} \to \begin{bmatrix} 1 & 3 & 3 & 3 \\ 1 & 7 & 1 & 9 \\ 1 & 7 & 8 & 2 \\ 5 & 6 & 2 & 9 \\ 9 & 1 & 2 & 3 \end{bmatrix}$

◄

► **1.6.** Write a function that takes an array of `n` time intervals, `(a,b)`, where `0<a<b`, and sorts them by:

- The earliest start.
- If tied, by shortest duration.

Example: $[(5, 7), (5, 6), (3, 4), (2, 8), (3, 6), (4, 5)] \to [(2, 8), (3, 4), (3, 6), (4, 5), (5, 6), (5, 7)]$ ◄

2 Theory Exercises

► **2.1.** Compute the sum $\sum_{j=k+m}^{n-p+2} 1$, where k, m, n, p are constants. ◄

► **2.2.** Compute the double sum $\sum_{i=k}^{n-3} \sum_{j=m}^{n^2} 1$, where k, m, n are constants. ◄

► **2.3.** In each of the following cases, write the given expression compactly using summation notation, $\sum$ (there is more than one correct answer for all cases).

1. $A = 4 + 9 + 16 + 25 + 36$;
2. $B = 2^2 + 3^4 + 4^6 + 5^8 + 6^{10}$;
3. $C = 2 - 3 + 4 - 5 + 6 - 7 + 8 - 9$;
4. $D = -4 + 6 - 8 + 10 - 12 + 14$. ◄

► **2.4.** Show that the complexity of sorting an array of n elements with insertion sort is $O(n^2)$. To show the upper bound of $O(n^2)$ consider an inversely sorted list. ◄

► **2.5.** Perform the final step of mergesort on the following sorted arrays:

$$2, 5, 8, 10 \text{ and } 7, 9, 11, 13, 15.$$ ◄

► **2.6.** Consider the following queue of playing cards.

Use bin sorting to sort this queue, so that the final queue of playing cards will be ordered first by suit (order is $\heartsuit, \diamondsuit, \spadesuit, \clubsuit$), and then by face-value (order is $A, 2, 3$). ◄

► **2.7.** (Non-examinable) Pigeonhole sort in-place the array $2, 4, 0, 1, 3$. ◄