

Homework 3

Assigned: October 7

Due: October 19

Please respect the following guidelines for writing pseudocode:

1. C instructions are fine. But do not write object-oriented additions. Do not declare or use any class. Declare only procedures (if necessary) and explain in words what each procedure does, and what is the use of each parameter.
2. One instruction per line
3. Match the brackets with a horizontal line
4. Number your lines
5. Write down if your array is indexed $0 \dots n - 1$ or $1 \dots n$.

Problem 1 Suppose we are given a sequence S of n elements, each of which is colored red or blue. Assuming S is represented as an array, give an in-place method for ordering S so that all the blue elements are listed before all the red elements. Can you extend your approach to three colors? In-place means that only swap operations are allowed.

Problem 2 Let S be an array with n integers. An *inversion* in S is a pair of elements x and y such that x appears before y in S but $x > y$. Describe an algorithm running in $O(n \log n)$ time for determining the *number* of inversions in S . Present pseudocode, and justify both correctness and running time.

Hint: Try to modify the merge-sort algorithm to solve the problem. Consider running merge-sort in parallel with your main algorithm

Problem 3 Characterize each of the following recurrence equations using the master method (assuming that $T(n) = c$ for $n < d$, for constants $d \geq 1$).

a. $T(n) = 2T(n/2) + \log n$

b. $T(n) = 8T(n/2) + n^2$

c. $T(n) = 16T(n/2) + (n \log n)^4$

d. $T(n) = 7T(n/3) + n$

e. $T(n) = 9T(n/3) + n^3 \log n$

Problem 4 Describe an efficient greedy algorithm for making change for a specified value using a minimum number of coins, assuming there are four denominations of coins (called quarters, dimes, nickels, and pennies), with values 25, 10, 5, and 1, respectively. Argue why your algorithm is correct.

Problem 5 Give an example set of denominations of coins so that a greedy change making algorithm will not use the minimum number of coins. Give an instance, show the output of the greedy algorithm on this instance, and show better output.