

Problem Set 4

Due: Friday, October 26

Reading: Handouts 12 (Array Sorting) and 15 (Linear Sorting); *CLRS* Chapters 7 and 8; *CLR* Chapters 8 and 9.

Suggested Problems: *CLRS* 8.1-1, 8.1-2; 8.2-3, 8.3-2; 8.3-3 (*CLR* 9.1-1, 9.1-2; 9-2.3; 9.3-2, 9.3-3).

Required Problems: You should write up and turn in the solutions to the problems listed below: The points awarded per problem are given in brackets.

Problem 1 [33]: The Dutch National Flag Problem

(The following problem was proposed by W.H.J. Feijen and made famous by Edsger W. Dijkstra, both of Dutch origin.)

You are given a row of n buckets, each containing one ball, which may be either red, white, or blue. Your goal is to rearrange the balls in the buckets such that they appear in the order of the colors on the Dutch national flag: red balls should be grouped on the left, white balls in the middle, and blue balls on the right. The only operation you may use to move balls is $\text{swap}(i, j)$, which swaps the contents of the i th and j th buckets.

- a. [15] Give pseudocode for a linear-time algorithm for sorting an array $B[1..n]$ of balls in the Dutch national flag order. Your algorithm should use only constant space in addition to the given array. Hint: study the Lomuto partitioning technique for quicksort from Handout 12 and *CLRS* 7.1 (unfortunately, *CLR* only covers Lomuto in problem 8-2).
- b. [15] Prove that your algorithm is correct by the method of loop invariants. Recall that this requires the following steps:
 1. State your loop invariants;
 2. Show that the loop invariants hold the first time the loop is entered;
 3. Show that if the loop invariants are true at the beginning of iteration i , they are true at iteration $i + 1$;
 4. Show that the loop terminates; and
 5. Show that the terminating state has the desired correctness properties.
- c. [3] According to the decision-tree analysis performed in class (and in *CLRS* 8.1/*CLR* 9.1), the best worst-case running time of a comparison-based sorting algorithm is $\Theta(n \cdot \lg n)$. Yet the Dutch national flag problem is a linear-time sorting algorithm. Explain how this can be.

Problem 2 [27]: Stooge Sort

Do *CLRS* Problem 7-3 (*CLR* Problem 8-3).

For part a, give a proof of correctness by induction on the number of elements $n = (j - i + 1)$ in the array. Recall the structure of an induction proof for the correctness of an algorithm:

- *Base Case(s)*: Prove that the algorithm is correct on the base case(s).
- *Inductive Case(s)*: Assuming that the algorithm is correct on all inputs whose size is $< n$, prove that the algorithm is correct on all inputs of size n .

The correctness proof is somewhat subtle, so please explain it carefully. For simplicity, you may assume that the array segment $A[i..j]$ is non-empty and that all elements in this array segment are distinct.

As part of your proof, you should carefully show that the arrays $A[i..j-k]$ and $A[i+k..j]$ mentioned in lines 6–8 have lengths less than n . You should also carefully show the bounds (as a function of n) on the sizes of the three array segments $A[i..i+k-1]$, $A[i+k..j-k]$, and $A[j-k+1..j]$.

Hint: For each element in the original segment $A[i..j]$, define its *fate* as one of the following three labels:

- *lo* if it would end up in the first segment $A[i..i+k-1]$ in the final sorted segment;
- *mid* if it would end up in the second segment $A[i+k..j-k]$ in the final sorted segment; and
- *hi* if it would end up in the third segment $A[j-k+1..j]$ in the final sorted segment.

Use these fate labels to guide your correctness proof.

Grading breakdown on parts of this problem:

- Part a: [15]
- Part b: [10]
- Part c: [2]

Problem 3 [20] Do *CLRS* 8.2-4 (*CLR* Exercise 9.2-5). Note that the notation $[a..b]$ means the range of integers from a to b , inclusive.

Problem 4 [20] Do *CLRS* Exercise 8.3-4 (*CLR* Exercise 9.3-4). *Hint*: View an integer x in the range $[1..n^2]$ as a pair of "digits" $(x \operatorname{div} n, x \operatorname{mod} n)$ (where $x \operatorname{div} n$ finds the integer quotient of x divided by n , and $x \operatorname{mod} n$ finds the integer remainder of x divided by n).

Extra Credit 1 [20]: Insertion Correctness

Show that the **Insert** procedure from Handout 12 is correct by the method of loop invariants using the following invariants (which are expressed in terms of an implicit index variable k):

insertLI1 $A_k[h+1..length(A_k)] = A_{init}[h+1..length(A_{init})]$.

insertLI2 $elts(A_k[1..j_k-1]) \cup_{\text{bag}} elts(A_k[j_k+1..h]) \cup_{\text{bag}} \{\text{key}\} =_{\text{bag}} elts(A_{init}[1..h])$

insertLI3 $sorted(A_k[1..h])$

insertLI4 $(j_k = h)$ or $\text{key} < A_k[j_k+1]$. (Alternatively, could use the stronger invariant: key is $<$ each element in $A_k[j_k+1..h]$.)

Note that invariants (insertLI3), (insertLI4), and (insertLI5) on Handout 12 have been replaced by a simpler single invariant (insertLI3); this improvement is due to Holly Muenchow. The invariant (insertLI4) above corresponds to (insertLI6) on Handout 12.

Extra Credit 2 [25]: Two-Finger Partition Correctness

Prove the correctness of the **Two-Finger-Partition** algorithm by fleshing out the missing details in the skeleton on p. 13 of Handout 12'.

Extra Credit 3: Sorting Shootout

Pick a programming language (perhaps your favorite; perhaps one you want to learn). In this language, implement all the list sorting algorithms from Handout 10 and/or all the array sorting algorithms from Handout 12. Perform experiments in which you measure the actual running time of the algorithms on randomly permuted lists/arrays of integers between 1 and n , for $n = 10^k$ where k ranges from 1 to 9. How well do the worst-case asymptotic running times predict the relative performance of the algorithms on the inputs for various k ?

The credit given for this problem depends on what you do; talk to Lyn for details/guidance.

Problem Set Header Page
Please make this the first page of your hardcopy submission.

CS231 Problem Set 4

Due Friday, October 26

Name:

Date & Time Submitted:

Collaborators (*anyone you worked with on the problem set*):

*In the **Time** column, please estimate the time you spend on the parts of this problem set. Please try to be as accurate as possible; this information will help me design future problem sets. I will fill out the **Score** column when grading your problem set.*

Part	Time	Score
General Reading		
Problem 1 [33]		
Problem 2 [27]		
Problem 3 [20]		
Problem 4 [20]		
Extra Credit 1 [20]		
Extra Credit 2 [25]		
Extra Credit 3		
Total		