CS1800 Discrete Structures	Prof Higger & Prof Hamlin
Fall 2024	Nov 22, 2024

HW9: Algorithms

Due: Dec 3, 2024 @ 11:59 PM

Instructions:

- HW instructions
- academic integrity and collaboration

Problem 1 [10 pts (2 each)]: Log Practice

Solve each of the following equations for x. Numbers are selected so that its possible to do each of these problems in your head, no calculator is needed.

i $x = \log_2 64$

ii $5 = \log_2 x$

iii $3 = \log_x 8$

iv $x = \log_2 \frac{1024}{17} + \log_2 17$

v $x = \log_2 1024^{17}$

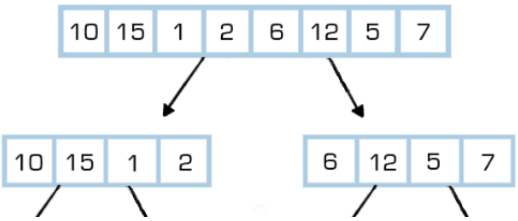
Problem 2 [20 pts]: Sorting Steps

Sort the following list with the following algorithms, showing all the intermediate steps.

$$18, 45, 23, 2, -5, 10, 99, 0$$

i Insertion Sort
(Rewrite the list at each step, using the □ symbol to divide the sorted portion of the array on its left from the unsorted portion to its right.)

ii Merge Sort
(Draw an array which shows the splitting and re-combining of the list. The first step, on another list of values, is shown below to demonstrate the intended notation)



Problem 3 [24 pts]: Three Stooges

Moe, Larry and Curly have just purchased three new computers, each with its own processing speed and sorting algorithm:

	Comparisons / sec	Search Algorithm	$T(n)$
Moe	50	Linear Search	n
Larry	5	Optimal Chunk Search	$2\sqrt{n}$
Curly	1	Binary Search	$\log_2 n$

where $T(n)$ is the number of comparisons it takes, in the worst case, to sort a list of size n.

Note that questions ii and iii below are not easily solved with pencil and paper, please show an initial equation and use Wolfram Alpha (https://www.wolframalpha.com/input?i=x%5E2+%2B+5x+%2B+6+%3D+17+x) to compute a final answer (the ”solutions” box on the linked page may be helpful). There is a very similar example in recitation10, whose solutions you may access, which may also be helpful here.

i What is the smallest list input size n (whole number) which ensures that, for any larger n and worst case list per method, Larry's computer sorts faster than Moe's?

ii What is the smallest list input size n (whole number) which ensures that, for any larger n and worst case list per method, Curly's computer sorts faster than Moe's?

iii What is the smallest list input size n (whole number) which ensures that, for any larger n and worst case list per method, Curly's computer sorts faster than Larry's?

Problem 4 [24 pts]: Recurrence

Solve eac of the following recurrences by substitution. Assume a base case of $T(1) = 1$. As part of your solution, you will need to establish a pattern for what the recurrence looks like after the k-th substitution. Check that this pattern is consistent with your substitutions, but you do not need to formally prove it is correct via induction.

i $T(n) = T(n-2) * 7$

Problem 5 [2 extra credit pts]: Bubble Sort Extra Credit

An array is *nearly-d sorted* if any element is not further than d spots from its sorted position. Consider the sorted list of elements:

$$X = [1, 5, 9, 10, 15, 20, 34, 57, 66, 91]$$

The same elements form a nearly-sorted list:

$$A = [1, 5, 10, 15, 9, 20, 34, 57, 91, 66]$$

with $d = 2$ because each value is, at most, 2 spots from its sorted position. Consider that the value 9 is out of order as it is in index 4[1] in A while it is in index 2 in X. Similarly,

$$B = [1, 5, 10, 9, 15, 20, 34, 57, 91, 66]$$

is nearly sorted with $d = 1$ as value 9 is in index 3 instead of index 2, 66 is in index 9 instead of index 8 and so on.

Bubble Sort[2], a sorting algorithm we have not covered, has an advantage over other methods when operating on nearly-d sorted lists.

i Describe Bubble Sort's Advantage in the best case scenario over other methods.

ii Bubble Sort need only pass through a nearly-d sorted list d times to ensure the list is sorted. Justify why this is the case. (Hint: consider the early termination condition of Bubble Sort)

[1] We adopt the Python convention of indexing $0, 1, 2, 3, \ldots$

[2] Wikipedia is a great place to start your Bubble Sort studying https://en.wikipedia.org/wiki/Bubble_sort, the animation in particular was instructive. However, more kinesthetic learners may appreciate the following video too: https://www.youtube.com/watch?v=lyZQPjUT5B4.