.

HGEN 48800 (2024 spring)
Problem Set #1

Part I : Xin's Lectures

Problem 1 Consider the ***searching problem***:

Input: A sequence of n numbers $A = (a_1, a_2, \cdots, a_n)$ and a value v.

Output: An index i such that $v = A[i]$ or the special value $NULL$ if v does not appear in A.

(a) (5 points) Implement the ***linear search*** algorithm, which scans through the sequence, looking for v. Using a loop invariant, prove that your algorithm is correct.

(b) (5 points) For linear search, how many elements of the input sequence need to be checked on the average, assuming that the element being searched for is equally likely to be any element in the array? How about in the worst case? What are the average-case and worst-case running times of linear search in Θ-notation? Justify your answers

(c) (5 points) If the sequence A is sorted, we can check the midpoint of the sequence against v and eliminate half of the sequence from further consideration. Implement the ***binary search*** algorithm either iteratively or recursively.

(d) (5 points) What is the worst-case running time of binary search in Θ-notation?

Problem 2 Using the ***master method***:

(a) (10 points) Use the master method to give tight asymptotic bounds for the following recurrences:

i. $T(n) = 2T(n/4) + 1$

ii. $T(n) = 2T(n/4) + \sqrt{n}$

iii. $T(n) = 2T(n/4) + n$

iv. $T(n) = 2T(n/4) + n^2$

(b) (10 points) Use the master method to show that the worst-case running time you gave in 1.d) for the binary-search recurrence is correct.

Problem 3 Based on the pseudo code taught in class, implement the ***merge-sort*** algorithm for the sorting problem:

Input: A sequence of n numbers $A = (a_1, a_2, \cdots, a_n)$.
Output: A reordered sequence $(a'_1, a'_2, \cdots, a'_n)$ such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$.

Part II : Mengjie's Lectures

Can you build a hash table for 3-tuples for the following sequences and search for string "TAGCTAGCT"? Write down the pseudo code and then implement it.

S1 = GCTGCTGCTGCTAAACGTTTGGGGCAGTCGAT
S2 = GGTGCTCCAAGCTTTTGAGTCTGCTAGTGTCAACCCT
S3 = GTGGGCCCCCTAGCTAGCTAGCTGGGGCAC
S4 = TGTCGCTGGCTGGACTGCTGATCGTAGTAG