COMP 6651: Worksheet 2

1. Use the divide-and-conquer integer multiplication algorithm to multiply two integers 83645283 and 75461934.

2. Suppose you are choosing between the following two algorithms:

   (A) This algorithm solves problems by dividing them into six subproblems of half the size, recursively solving each subproblem, and then combining the solutions in linear time.

   (B) This algorithm solves problems of size n by recursively solving three subproblems of size $n-1$ each and then combining the solutions in constant time.

   (a) Write down the recurrence for the running time of each algorithm.

   (b) Solve the recurrences.

   (c) Which algorithm would you choose and why?

3. Consider the following procedure:

---

procedure $F(n)$
    if $n > 1$ **then**
        print line "still printing..."
        $F(n/2)$
        $F(n/2)$
        $F(n/2)$

   (a) Write down the recurrence for the number of lines that this procedure prints. You may assume that n is a power of 2.

   (b) Solve the recurrence.

4. The following divide-and-conquer algorithm finds the maximum value in array $A[1..n]$. The initial call is to $Maximum(A, 1, n)$.

---

procedure $Maximum(A, p, r)$
    if $r - p \leq 1$ **then**
        return $\max(A[p], A[r])$
    else
        $q \leftarrow \lfloor (p+r)/2 \rfloor$
        $m_1 \leftarrow Maximum(A, p, q)$
        $m_2 \leftarrow Maximum(A, q+1, r)$
        return $\max(m_1, m_2)$

   (a) Prove the algorithm is correct. You may assume n is a power of 2.

   (b) Write down a recurrence for the worst-case number of comparisons used on arrays of length n.

   (c) Solve the recurrence.

5. You are given an infinite array A that contains n integers in the first n positions (indexing starts with 1) and ∞ in all other positions. You do *not* know the value of n. Your task is to design an iterative algorithm that finds the value of n using $O(\log n)$ accesses to array A.

(a) Describe your algorithm in plain English (maximum 5 short sentences).

(b) Describe your algorithm in pseudocode.

(c) For each loop in your algorithm, state a useful loop invariant (without proof). State the corresponding termination condition(s) and how correctness follows from the termination condition(s).

(d) Argue that the running time is $O(\log n)$ (as measured by the number of accesses to array A). Mention the running time of each logical block of your algorithm.

6. You are given an array A of n images. Some of these images might be identical. For $i \neq j \in [n]$ you can invoke a comparison procedure that returns whether the images $A[i]$ and $A[j]$ are identical or not. This procedure is denoted by $A[i] == A[j]$. Design a divide and conquer algorithm to decide whether there is an image that appears more than $n/2$ times in A using $O(n)$ invocations of the comparison procedure. Solutions using asymptotically more invocations of the comparison procedure receive the grade of 0.

   (a) Describe your algorithm in plain English (maximum 5 short sentences).

   (b) Describe your algorithm in pseudocode.

   (c) Provide a concise argument of correctness of your algorithm.

   (d) State the recurrence of the number of invocations of the comparison procedure (do not forget the base case).

7. Given a sorted array of *distinct* integers $A[1..n]$, you want to find out whether there is an index i for which $A[i] = i$. Design a decrease-and-conquer (special case of divide-and-conquer with a single subproblem solved recursively) algorithm that runs in $O(\log n)$ time.

   (a) Describe your algorithm in plain English (at most 5 short sentences).

   (b) Describe your algorithm in pseudocode.

   (c) State a useful loop/recursion invariant (without proof). State the termination condition(s) and how it implies correctness.

   (d) Prove that the running time of your algorithm is $O(\log n)$ (you may use Master's theorem). Make sure any notation you introduce is defined precisely.

8. The Hadamard matrices $H_0, H_1, H_2, \ldots$ are defined as follows:

   - H_0 is the 1×1 matrix $\begin{bmatrix} 1 \end{bmatrix}$,
   - For $k > 0$, H_k is the $2^k \times 2^k$ matrix $H_k = \left[\begin{array}{c|c} H_{k-1} & H_{k-1} \\ \hline H_{k-1} & -H_{k-1} \end{array}\right]$

   Your goal is to design a divide-and-conquer algorithm for the following problem. Your algorithm should use $O(n \log n)$ single-digit operations: multiplications, additions, subtractions.

   Input: v — a column vector of length $n = 2^k$

   Output: $H_k v$ — the product of the Hadamard matrix with v

   (a) Describe your algorithm in plain English. Briefly justify correctness.

   (b) Describe your algorithm in pseudocode.

(c) State the recurrence for the number of single-digit operations performed by your algorithm. State what the recurrence solves to. Do not forget the base case!

9. You are given an array A with n entries that are distinct integers. You may assume that $n \geq 2$ and that n is a power of 2. Moreover, you know that A is *unimodal*: there exists an index p such that $A[1], A[2], \ldots, A[p]$ is an increasing sequence and $A[p], A[p+1], \ldots, A[n]$ is a decreasing sequence. You would like to find the "peak entry index" p by accessing only a small number of entries of A. Design a divide and conquer (or decrease and conquer) algorithm to find p by reading at most $O(\log n)$ entries of A.

   (a) Describe your algorithm in plain English. Briefly justify correctness.

   (b) Describe your algorithm in pseudocode.

   (c) Analyze the runtime of your algorithm by stating a relevant recurrence for the number of entry lookups performed by your algorithm. State what the recurrence solves to. Do not forget the base case!

10. Show how to multiply the complex numbers $a + bi$ and $c + di$ using only three multiplications of real numbers. The algorithm should take a, b, c, d as input and produce separately the real part $ac - bd$ and $ad + bc$.