1. Consider the following program (which doesn’t do anything interesting):

Procedure Split\((A, L, n)\)

\[
\text{for } i = 1 \text{ to } n \\
\quad \text{increment } A[L + i - 1] \\
\text{end for} \\
\text{if } n > 1 \text{ then} \\
\quad m = \lfloor n/3 \rfloor \\
\quad \text{Split}(A, L, m) \\
\quad \text{Split}(A, L + m + 1, m) \\
\quad \text{Split}(A, L + 2m + 1, n - 2m) \\
\text{end if end}
\]

(a) Write a recurrence for the running time \(T(n)\) of procedure Split on input array \(A\) of length \(n\), integer \(L\) initially 1, and integer \(n\).

(b) What is solution of the recurrence you found using \(O\) notation? (You may assume that \(n\) is a power of 3.) (You do not need to justify your answer.)

2. (20 points) Consider the following program:

Procedure Slowzero\((A, n)\)

\[
\text{for } i = 1 \text{ to } n - 1 \\
\quad A[i] \leftarrow 0 \\
\quad \text{Slowzero}(A, i) \\
\text{end for}
\]

(a) Write a recurrence for the running time \(T(n)\) of the Slowzero procedure on input \(A\) and \(n\) where \(A\) is of length \(n\). (You may assume that the array \(A\) is referenced by name in the recursive calls so that it is never copied.)

(b) Prove that \(T(n) \leq c2^n - d\) for some constants \(c\) and \(d\) and all \(n \geq 1\).

3. (15 points) Give an expression using Big-O notation that describes the behaviour of the following recurrences as closely as you can. (All fractions are assumed to be rounded down to the nearest integer.)

(a) For \(n \geq 2\), \(T(n) \leq 2T(n/2) + cn; T(1) = 1\).

(b) For \(n \geq 3\), \(T(n) \leq T(n - 2) + n; T(1) = T(2) = 1\).

(c) For \(n \geq 2\), \(T(n) \leq 3T(n/2) + cn; T(1) = 1\).

(d) For \(n \geq 2\), \(T(n) \leq 2T(n/2) + cn^3; T(1) = 1\).

(e) For \(n \geq 2\), \(T(n) \leq 4T(n/2) + cn^2; T(1) = 1\).
4. Suppose that $T(n) = T(n/4) + T(3n/4) + n$ for $n \geq 4$ and $T(n) = 0$ for $n < 4$. (All fractions are assumed to be rounded down to the nearest integer.)
Prove that $T(n) \leq cn \log_2 n$ for some constant $c$.
Hint: Use guess and verify. Guess that it works and figure out how big $c$ must be. Note that $\log_2(n/4) = \log_2 n - 2$ and $\log_2(3n/4) < \log_2 n - 1/3$.

5. Design a simple divide and conquer algorithm to compute $x^n$ using a very small number of multiplications.
Write a recurrence to describe the number of multiplications your algorithm uses as a function of $n$. Write out the solution of this recurrence. (You can use big-O notation.)

6. (a) Design an algorithm to take two strings $a = a_1a_2\cdots a_m$ and $b = b_1b_2\cdots b_n$, input in arrays $A$ and $B$ respectively, and determine the least number of delete and insert operations needed to transform $a$ into $b$ where each operation adds or removes exactly one character.
(b) Analyse the running time of your algorithm.
(c) Give a simple calculation using the result of the previous algorithm to determine the length of the longest common subsequence in $a$ and $b$. That is, determine the largest $\ell$ such that there are indices $1 \leq i_1 < i_2 < \cdots < i_\ell \leq m$ and $1 \leq j_1 < j_2 < \cdots < j_\ell \leq n$ so that $a_{i_1}a_{i_2}\cdots a_{i_\ell} = b_{j_1}b_{j_2}\cdots b_{j_\ell}$.