

1. Suppose a computer with 10 processors can do 90 percent of the work 10 times faster than a one processor computer but the 10 processors take the same time for the remaining 10 percent of the work. Over all, how much faster is the 10 processor computer?
2. Approximate $\sum_{1 \leq i \leq n} n^{3/2}$.
3. Solve the recurrence $T_{2n+1} = 3T_n + an + b$.
4. Solve the recurrence $T_n = T_{n-1} + T_{n-3} + 1$.
5. Suppose Algorithm 5.2 is modified so that $T_1 = (U_1 + U_2) \bmod 2^n$, $d_1 = \lfloor (U_1 + U_2)/2^n \rfloor$, $T_2 = (U_1 + U_2) \bmod 2^n$, $d_2 = \lfloor (U_1 + U_2)/2^n \rfloor$.
 - 5a. What additional changes are needed to make the algorithm be correct?
 - 5b. Write the exact recurrence for this modified algorithm. You need be concerned only with the case where the original number as an even number of bits.
6. Algorithm 5.2 was designed for multiplying two number that each have $2n$ bits. Modify the algorithm to multiply two numbers with $2n + 1$ bits. In your algorithm U_1 and V_1 should each have $n + 1$ bits, while U_2 and V_2 are unchanged (n bits).
7. A substring of string A is made by removing zero or more characters from A and closing in the holes. In other words, if A is a string of length n and B is a substring of length m then
 - (a) $B[i] = A[j_i]$ for $1 \leq i \leq m$ (each position in B corresponds to a position in A that has the same character) and
 - (b) $i_1 > i_2$ implies that $j_{i_1} > j_{i_2}$ (the positions occur in the same relative order).Given two strings X and Y , the longest common substring problem is to find the longest string Z such that Z is a substring of X and Z is also a substring of Y .
 - 7a. Why is this problem a dynamic programming problem? What is the key insight that can be used to design a fast algorithm for this problem?
 - 7b. Sketch the outline of an algorithm to solve this problem quickly. In other words, you need to give the basic design of the algorithm but you do not need to express all the steps with full algorithmic precision.