Problem 1-1 (Insertion Sort Using a Linked List) 10 points

Consider the problem of implementing insertion sort using a doubly-linked list instead of array. Namely, each element $a$ of the linked list has fields $a.previous$, $a.next$ and $a.value$. You are giving a stating element $s$ of the linked list (so that $s.previous = nil$, $s.value = A[1]$, $s.next.value = A[2]$, etc.)

(a) (8 points) Give a pseudocode implementation of this algorithm, and analyze its running time in the $\Theta(f(n))$ notation. Explain how we do not have to “bump” elements in order to create room for the next inserted elements. Is this saving asymptotically significant?

(b) (2 points) Can we speed up the time of the implementation to $O(n \log n)$ by utilizing binary search?

Problem 1-2 (Multiplying Binary Integers) 10 Points


(a) (2 points) Prove that $c = \sum_{(i:B[i]=1)} (a \cdot 2^i)$.

(b) (4 points) Write an $O(n+i)$ time procedure $\text{SHIFT}(A,n,i)$ to compute the $(n+i)$-bit product $a \cdot 2^i$.

(c) (4 points) Assume you are given $O(n)$ procedure $\text{ADD}(X,m,Y,k)$ which adds an $m$-bit $X$ to $k$-bit $Y$, where $m,k \leq 2n$. Using this procedure, and your work in parts (a) and (b), write the pseudocode to produce the desired product array $C$. Analyze the running time of your procedure in the $\Theta(f(n))$ notation, for appropriate function $f(n)$.

Problem 1-3 (Polynomial Evaluation) 16 (+4) Points

A degree-$n$ polynomial $P(x)$ is a function

$$P(x) = a_0 + a_1x + \ldots + a_{n-1}x^{n-1} + a_nx^n = \sum_{i=0}^{n} a_ix^i$$

PS1, Page 1
(a) (2 points) Express the value $P(x)$ as

$$P(x) = a_0 + a_1 x + \ldots + a_{n-2} x^{n-2} + b_{n-1} x^{n-1} = \sum_{i=0}^{n-1} b_i x^i$$

where $b_0 = a_0, \ldots, b_{n-2} = a_{n-2}$. What is $b_{n-1}$ as a function of the $a_i$’s and $x$?

(b) (5 points) Using part (a) above write a recursive procedure $\text{Eval}(A, n, x)$ to evaluate the polynomial $P(x)$ whose coefficients are given in the array $A[0 \ldots n]$ (i.e., $A[0] = a_0$, etc.). Make sure you do not forget the base case $n = 0$.

(c) (3 points) Let $T(n)$ be the running time of your implementation of $\text{Eval}$. Write a recurrence equation for $T(n)$ and solve it in the $\Theta(\cdot)$ notation.

(d) (6 points) Assuming $n$ is a power of 2, try to express $P(x)$ as $P(x) = P_0(x) + x^{n/2} P_1(x)$, where $P_0(x)$ and $P_1(x)$ are both polynomials of degree $n/2$. Assuming the computation of $x^{n/2}$ takes $O(n)$ times, describe (in words or pseudocode) a recursive procedure $\text{Eval}_2$ to compute $P(x)$ using two recursive calls to $\text{Eval}_2$. Write a recurrence relation for the running time of $\text{Eval}_2$ and solve it. How does your solution compare to your solution in part (c)?

(e) (Extra Credit.) Explain how to fix the slow “conquer” step of part (d) so that the resulting solution is as efficient as “expected”.

Problem 1-4 (Asymptotic Comparisons) 10 Points

For each of the following pairs of functions $f(n)$ and $g(n)$, state whether $f$ is $O(g)$; whether $f$ is $o(g)$; whether $f$ is $\Theta(g)$; whether $f$ is $\Omega(g)$; and whether $f$ is $\omega(g)$. (More than one of these can be true for a single pair!)

(a) (2 points) $f(n) = 15n^{15} + 2; \ g(n) = \frac{n^{16} + 3n^2 + 4}{11} - 37n$.

(b) (2 points) $f(n) = \log(n^{11} + 3n); \ g(n) = \log(n^2 - 1)$.

(c) (2 points) $f(n) = \log(2^{n} + n^{12}); \ g(n) = \log(n^{12})$.

(d) (2 points) $f(n) = n^5 \cdot 2^n; \ g(n) = n^2 \cdot 3^n$.

(e) (2 points) $f(n) = (n^n)^{10}; \ g(n) = n^{n^2}$.