CSE 326 Problem Set 2 Due: October 20, 1997

1. Let us say that function $f(n)$ grows no faster than $g(n)$ if $f(n) = O(g(n))$, and that $f(n)$ grows at the same rate as $g(n)$ if $f(n) = \Theta(g(n))$. Arrange the following list of functions in increasing order of growth rate and indicate which functions have the same growth rate: $n, \sqrt{n}, n^{1.5}, n^2, n \log n, 2^n, 10n^2 + n, 37, n \log\log n, n^2 \log n, n^{\log_2 n}, n \log(n^2), n^{\log_3 n}$

2. Suppose $T_1(n) = O(f(n))$ and $T_2(n) = O(f(n))$. Which of the following must be true?

   (a) $T_1(n) + T_2(n) = O(f(n))$;

   (b) $T_1(n) - T_2(n) = o(f(n))$;

   (c) $\frac{T_1(n)}{T_2(n)} = O(1)$;

   (d) $T_1(n) = O(T_2(n))$.

3. Consider the pseudocode given below for the sorting algorithm Slowsort. Here A is an array of integers and i and j are indices into the array. Slowsort(A, i, j) sorts the integers between positions i and j in array A.

   (a) Prove that Slowsort works correctly.

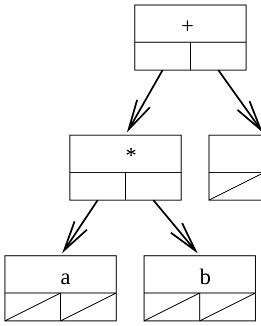
   (b) Write a recurrence relation for the running time of Slowsort as a function of $j - i + 1$, the size of the array to be sorted.

   (c) Solve the recurrence relation.

```
     Slowsort(i,j,A)
if i=j then return
 else
  Slowsort(i+1,j, A);
  if A[i] > A[i+1] then swap A[i] with A[i+1];
  Slowsort(i+1,j, A)
```

4. Let T be a binary expression tree. Give pseudocode for an algorithm that computes the parenthesized expression corresponding to the tree. Your output should be a linked list, in which the data fields of the nodes contain the successive symbols of the expression. An example of input and output are given below. Give a brief explanation of why your algorithm works.

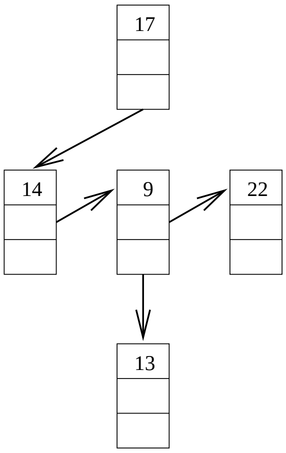
INPUT:



OUTPUT: (a * b) + c

5. You are given a tree in which each node has a data field, a Next Sibling field and a First Child field. Give pseudocode for an algorithm which computes a linked list giving the data fields of the nodes in increasing order of the depths of the nodes (where the depth of a node is its distance from the root). An example of input and output are given below. Give a brief explanation of why your algorithm works.

INPUT:



OUTPUT: