CS 692 Capstone Exam Algorithms Spring 2023.

Choose any 2 of the 3 problems.
If you answer all three questions, only questions 1 and 2 will be graded.

Full name: ________________ Net ID:________________

Question 1) (20 points)
Consider the following recurrence relations. Express each in Big-O(). Show all your work. You can use the Master Theorem (if applicable) or any other technique. $T(1)=1$ in all the cases. **(5 points each)**

A) $T(n) = T(n/2) + 3n$

B) $T(n)= 2T(n/2)+ n \log n$

C) $T(n)=9T(n/3)+ O(1)$

D) $T(n)=T(n-1)+ 1$

Full name: _________________ Net ID:_________________

Question 2) (20 points)
Part 1) (12 points) Take the following list of functions and arrange them in ascending order of growth rate. That is, if function g(n) immediately follows function f(n) in your arrangement, then it should be the case that f(n) is O(g(n)). No justification is needed.

$f1(n)= 2^n$
$f2(n)= n^4$
$f3(n)= n$
$f4(n)=100^n$
$f5(n)=n \log n^4$
$f6(n)= \log n$
$f7(n)= n^n$
$f8(n)=n!$

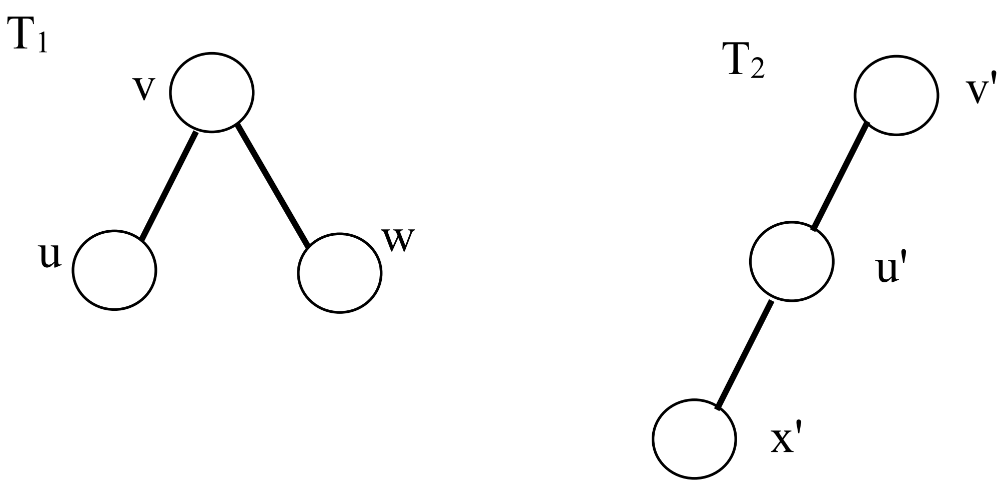
Part 2) (4 points each) Let two functions f(n) and g(n) reflect the total number of basic operations in two algorithms A_1 and A_2, respectively.

A) Assume f(n)= little-o (g(n)). What will be the result of $\lim_{n\to\infty} \frac{f(n)}{g(n)} =$? Justify your answer in at most 5 sentences. Be precise.

B) Assume f(n)=Big-O(g(n)). What will be the result of $\lim_{n\to\infty} \frac{f(n)}{g(n)}$ =? No justification is needed here.

Full name: ____________________ Net ID:____________________

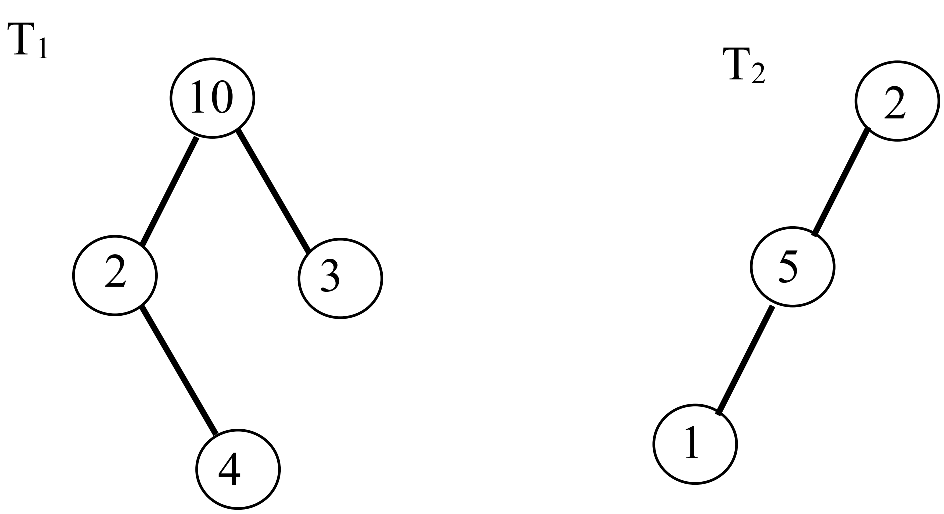
Question 3) (C/C++ coding question) (20 points)
Consider two binary trees T_1 and T_2. Assume nodes in both trees are labeled with integer numbers.
Definition: We say two nodes in trees T_1 and T_2 **overlap** if the node in T_1 is in the same position (same level and being left or right) as the node in T_2.
For example, in the below figure, the node $v \in T_1$ overlaps with the node $v' \in T_2$, and the node $u \in T_1$ overlaps with $u' \in T_2$. However, the node $w \in T_1$ does not overlap with any node in T_2. Also, the node $x' \in T_2$ does not overlap with any node in tree T_1.



Write a **recursive** Magic function that receives pointers to the roots of trees T_1 and T_2 and returns a pointer to the root of a newly constructed tree, called T_3, where the nodes in T_3 are going to be constructed as follows:

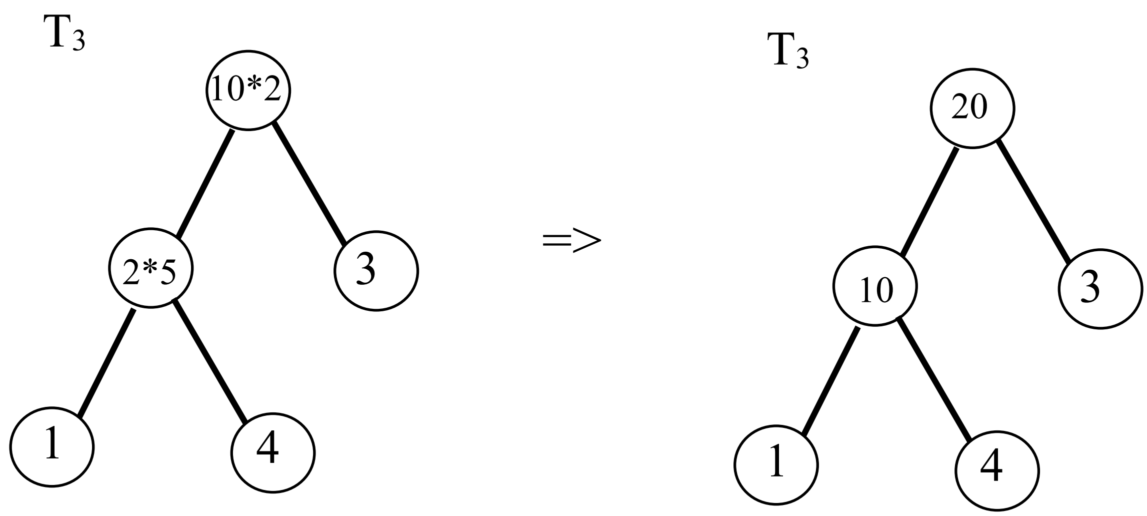
1) If two nodes in trees T_1 and T_2 overlap, the product of their labels will make the label for the corresponding node in tree T_3.
2) Otherwise, the non-null node label will be used for labeling the node in tree T_3.

For example, let tree T_1 and T_2 be as follows:



Full name: __________________ Net ID:__________________

The new tree T_3 will look like this:



Again, the input to the function is a pointer to the root of (possibly empty) tree T_1 and a pointer to the root of (possibly empty) tree T_2 and it returns a pointer to the root of tree T_3.
All trees should be implemented using **singly linked lists**.

A) **(4 points)** Declare your data structure.

B) **(10 points)** Write a C/C++ code for the Magic **function** as described above (a non-recursive function will receive 0 points. Code only in C or C++).

C) **(6 points)** Analyze the time complexity of your Magic function in the worst case, assuming that tree T_1 has n_1 nodes and tree T_2 has n_2 nodes. Explain your answer.