CSC 6013 Week 6 – Recursive algorithms and recurrence relations

Submit your work – codes, outputs, and asymptotic analyses - as a single docx or pdf file through Canvas.

Topics: Coding a recursive function; recurrence relation; back substitution or master method

1) a) Write python code for a recursive algorithm that will calculate the number of digits in the binary expansion/representation of a positive integer n. The logic of the recursive algorithm should be something like:

> if n = 1, the answer is 1;
>
> if n > 1, the answer is 1 more than the number of digits in the binary representation of n/2.

You might want to use the python function math.floor() in your code.

Your function should have only one input parameter – a positive integer n. It should return the number of digits using a return statement. There should be no print statements in the function.

Note: Your function is determining the number of digits in the binary expansion of n. It is not creating the binary expansion of n.

Be sure to include comments identifying the input and output for this algorithm, as well as comments explaining what is accomplished by the key steps of the algorithm.

b) Run your code on the problem instances n = 256 and n = 750. Show me the output generated by having your main/driver code block call the function inside a print statement.

c) Create a recurrence relation that gives the work performed by the algorithm in the worst-case for a problem of size n. In your recurrence relation, count the number of recursive calls to the function as the fundamental unit of work. In your asymptotic analysis, you can assume that n is an integer power of 2.

d) Perform the asymptotic analysis with either the back-substitution method OR the master method to solve the recurrence relation and determine the algorithm's asymptotic class Big-Oh.

2) a) Write python code for a recursive algorithm that will calculate the sum of the squares of the positive integers $1^2 + 2^2 + 3^2 + ... + n^2$ when supplied with a positive integer n.

The logic of the recursive algorithm should be something like:

if n = 1, the answer is 1;

if n > 1, the answer is (the sum of the squares of the integers from 1 to n-1) + n^2.

Do not find a closed form formula for the summation; make your algorithm do all the arithmetic.

Your function should have only one input parameter – a positive integer n. It should return the sum of the sqaures using a return statement. There should be no print statements in the function.

b) Run your code on the problem instances n = 12 and n = 20. Show me the output generated by having your main/driver code block call the function inside a print statement.

c) Create a recurrence relation that gives the work performed by the algorithm in the worst-case for a problem of size n. In your recurrence relation, count the number of recursive calls to the function as the fundamental unit of work.

Note: You are tracking the number of recursive calls, not the result of the calculation.

d) Use the back-substitution method to solve the recurrence relation.