function f(x,n)
{
    if (x > 0)
    {
        f(x/2,n);
        for (var i from 0 until n)
        {println(n);}
    }
}

So I'm thinking the space complexity is theta(1).

Because the recursive call occurs X/2 times occupying x/2 space, but the for loop that executes after all the recursive calls runs n times, but does not use n space. Correct?

Subject: Re: Question 80 from PreReq Examination Material
Posted by jarobinson3 on Mon, 23 Jan 2017 20:01:58 GMT

You do not do X/2 calls, recall you are dividing X by 2 each time. The loop does not use any space. I am not sure of the space complexity because it depends on if you use the stack space used by the recursive call.

Subject: Re: Question 80 from PreReq Examination Material
Posted by lusth on Mon, 23 Jan 2017 21:27:12 GMT

You can assume space for the recursive call is allocated on the stack and that when the recursive call returns, that space is automatically freed.