CSC510 Homework 1

Chris Camano: ccamano@sfsu.edu

September 16, 2022

1. (a) Using the formal definition of big-O notation, show that

$$7n^4 - 21n^3 - 12n^2 + 13n + 91 = \mathcal{O}(n^4)$$

What are the values of C and K you used?

$$7n^4 - 21n^3 - 12n^2 + 13n + 91 \leq 7n^4 + 91 + 13n$$
$$7n^4 - 21n^3 - 12n^2 + 13n + 91 \leq 7n^4 + 91n^4 + 13n^4$$
$$7n^4 - 21n^3 - 12n^2 + 13n + 91 \leq 111n^4$$

Let C=111. Upon examination of the behavior of these two functions it appears that f(n) is $\mathcal{O}(g(n))\ \forall n \geq k = 1$ due the polynomial nature of the function

(b) Using the various identities regarding big-O notation, give a big-O estimate O(f(n)) for the function:

$$(3 + log_7(n))^2(17n^2 + log_8(n)) + 29(40n + 28)(25nlog_3(n) + 6n^2ln(n))$$

$$(3 + log_7(n))^2(17n^2 + log_8(n)) + 29(40n + 28)(25nlog_3(n) + 6n^2ln(n))$$
$$\mathbb{O}(log^2(n))\mathbb{O}(n^2) + \mathbb{O}(n)\mathbb{O}(n^2log(n))$$
$$\mathbb{O}(n^2log^2(n)) + \mathbb{O}(n^3log(n))$$
$$\mathbb{O}(n^2log^2(n) + n^3log(n))$$
$$\mathbb{O}(n^3log(n))$$

2. Convert the following Java method into pseudocode : Let me know if in the future you would like me to omit the end fors and end ifs here, they are auto populated by the latex environment I am using but I can toggle them going forward if that is your preference. thanks

3. Give big-O estimates for the return value count in each of these two functions:

Algorithm 1 SelectionSort(A[1,...,n])

```
1: for i=1,...,n-1 do
2:     minIndex:=i
3:     for  j=i+1,..,n do
4:         if A[j]<A[minIndex] then
5:             minIndex:=j
6:         end if
7:     end for
8:     swap A[minIndex], A[i]
9: end for
```

(a) For the first function the big O estimate is $\mathbb{O}(n^3)$ since there are three nested for loops acting as a bottle neck over the additional for loop at the end of the routine.

(b) For the second function the big O estimate is $\mathbb{O}(mn^3)$ since a function with time complexity $\mathbb{O}(n^3)$ is wrapped in another loop of complexity $\mathbb{O}(m)$ and we have the property that:

$$\mathbb{O}(f(n))\mathbb{O}(g(n)) = \mathbb{O}(f(n)g(n))$$

4. Prove, using induction, that for every n 1, our algorithm produces a list of instructions that:

   (a) starts with moving disc 1,

   (b) alternates between moving disc some other disc and moving disc 1, and

   (c) ends with moving disc 1.

   Let our inductive hypothesis be the case that all three of these statements are true.
   Base case: n=1
   In the event of the base case we simply move one disc from the start to the finish and all three conditions are satisified as there is no alternate disk to move anyways.
   Inductive step: For any given n The process of calling H(n) can be seen as three seperate parts. The first part is the recursive call which by the inductive hypothesis starts and ends at moving disc 1. The second part is moving the arbitrary disc for a given n. Finally the third step is the other recurisve call which again by the inductive hypothesis starts and ends with movign disc one.

   Due to the fact that each subroutine ends and starts with one, whenever it is the case that we move another disc this implies that we first move disc 1 and do so after as all steps except for the base case of n=0 can be viewed through this perspective. Thus by the process of mathematical induction we have proven the implication.