Homework 6 for ECS 20

Tamim Nekaien (915803826)

November 18, 2019

This assignment was made with $\heartsuit$ (and LaTeX).

Contents

1 Question 1

For B: The first result in the set of Natural numbers is $1 \in CUBE$
For I: $x + 3x^{2/3} + 3x^{1/3} + 1$
For E: no other $x \in CUBE$.

POST SOLUTION CHECKING CORRECTION: The I step can be rewritten as $(x^{\frac{1}{3}} + 1)^3$

2 Question 2

$|| x - 3 | -5 |= 10 =| x - 3 |= 15, -5$
Since absolute value cannot be negative, the second situation is not considered.
$| x - 3 |= 15 = x = 18, -12$
-12 and 18 are the two solutions for x.

POST SOLUTION CHECKING CORRECTION: Forgot intermediate step $| x - 3 |\geq 5$

3 Question 3

Proof by Induction:
Base Case: $n = 2, L_2 = L_1 + L_0 = 2 + 1 = 3$ and $L_2 = F_1 + F_3 = 1 + 2 = 3$
Inductive Step: $L_k = F_{k-1} + F_{k+1}$ and $L_k = L_{k-1} + L_{k-2}$
Use definition of Fibonacci numbers: $L_{k+1} = F_{k+1} + F_{k-1} + F_{k-2} + F_k$ and $L_{k-1} + L_{k-2} = F_{k+1} + F_{k-1} + F_{k-2} + F_k$
$\therefore$ The Lucas numbers hold true for k+1 terms in the given function using Fibonacci numbers.

4 Question 4

4.1 part a

The print statement: It is called on ixj times, which is $\left(\frac{n^2+n}{2}\right)$
BIg O: To find Big O you need to look at the highest power term: in this

case it's $O(n^2)$

4.2 part b

The print statement: It depends on whether the number is even or odd.
If even: The statement is run $n = 2^x$ times, or $log(n) times$
If odd: The statement runs an extra time because of the ceiling, so $log(n)+1$
Big O: The big O for this is $O(log(n))$.

5 Question 5

First proof: $\frac{x^3+7x^2+3}{2x+1} \leq Cx^2$
Raise all terms to highest power and solve-¿ $\frac{x^3+7x^3+3x^3}{2x} \leq Cx^2$
$\frac{11x^2}{2} \leq Cx^2$
Then $C = \frac{11}{2}, k = 1$
Second: $x^2 \leq C\frac{x^2+7x^2+3}{2x+1}$.
Express highest power: $\frac{x^3}{4x}$ $or \frac{1}{4}x^2$
So C $= \frac{1}{4}$ and k $= 1$

POST SOLUTION CHECKING CORRECTION*= I didn't show the relationship between x and k, $x > k$, and $x > 1$.

6 Question 6

Given some n to calculate:
int first = 1;
for (int i = 1; i ¡ n; i++)
first = first*(i);

The psuedocode will multiply n times.

POST SOlUTION CHECKING CORRECTION*= It's n-1, because i never reaches n, so there are n-1 calculations. And I could've just used a while loop instead.