Questions

1. $2n^2 + 4 < Cn^2$

   $2 + \frac{4}{n^2} < C$

   If n = 1 which will form the largest number possible:

   $2 + 4 < C$

   Therefore:

   C = 7

2. $3n^2 + 5n + 9 < Cn^3$

   $\frac{3}{n} + \frac{5}{n^2} + \frac{9}{n^3} < C$

   If n = 1 which will form the largest number possible:

   $3 + 5 + 9 < C$

   Therefore

   $C = 18$

Data Structures & Algorithms Labs
Week 2: Complexity and Efficiency

These exercises will allow you to test your knowledge of the lecture material from week 1 (complexity, efficiency, and search).

1 Programming Exercises

Once you have solved each of these exercises, try to figure out the (average/worst case) time and space complexity of your solution, and whether more efficient solutions exist.

► **1.1.** You are given a 2D integer matrix, that is an array of arrays of integers. It has size `m*n` where `m` is the number of arrays in the matrix and `n` is the size of each array (`matrix[i]`). Each array in the matrix is sorted in non-decreasing order. The arrays are also non-overlapping, in the sense that the last element of each array is not greater than the first element of the next (`matrix[i][m-1]` $\leq$ `matrix[i+1][0]` for all `i` in range). Create a function that that takes as input the given matrix and an integer target, and will return whether the matrix contains the target. ◄

► **1.2.** (Hard) repeat the previous exercise, but now do not assume that the arrays cannot overlap. Instead assume that both the rows and columns of the matrix are sorted, such that `matrix[i][j]` $\leq$ `matrix[i][j+1]` and `matrix[i][j]` $\leq$ `matrix[i+1][j]` for all `i` and `j` in range. ◄

► **1.3.** Given an integer x, $0 \leq x \leq 99999$, find its cube root without directly computing it (you can use Java's Math.pow function but only with integer exponent). Your answer must be within 10^{-3} of the exact answer. What if you instead allow x to be any floating point number in the given range? ◄

► **1.4.** (Hard) You are given an array of integers `a[0]`, ... `a[n-1]` and a positive integer `x`. You are allowed to perform the following operation as many times as you like: increase any element `a[i]` in the array by a positive integer amount. However, the sum of array elements cannot increase by more than `x`. Your goal is to maximise the smallest value in the array after performing the operations. Output the maximum possible value of the smallest element. ◄

2 Theory Exercises

In this section, the domain of all given functions is the set of positive integers: $\mathbb{Z}_+ = \{1, 2, 3, \ldots\}$. The following table evaluates some important functions of n for the first few values of n.

n	$\log_2 n$	$n \log_2 n$	n^2	n^3	2^n	n^n
1	0.00	0.00	1	1	2	1
2	1.00	2.00	4	8	4	4
3	1.58	4.75	9	27	8	27
4	2.00	8.00	16	64	16	256
5	2.32	11.61	25	125	32	3125
6	2.58	15.51	36	216	64	46656

A useful relation between different bases of logarithm is the following. Let $a, b, n > 0$. Then

$$\log_a n = \log_a b \cdot \log_b n.$$

To compare the growth of functions we often use the **Big-O notation**. This is defined as

$$f(n) \in O(g(n)) \iff \exists C, n_0 > 0 \text{ such that } \forall n \geq n_0 : f(n) \leq C \cdot g(n),$$

which (more verbosely) says that $f(n)$ is of order $g(n)$ if there exist two positive constants C and n_0 such that $f(n)$ is less than or equal to $Cg(n)$ for all n greater than or equal to n_0.

To show that $f(n) \in O(g(n))$ for a given pair of functions f, g, we need to find a pair of constants C, n_0, that satisfy the above definition. The pair C, n_0, need not be unique.

► **2.1.** In each of the following cases, find a constant $C > 0$ such that $f(n) < Cg(n)$ for all $n > 0$:

1. $f(n) = 2n^2 + 4$, and $g(n) = n^2$;
2. $f(n) = 3n^2 + 5n + 9$, and $g(n) = n^3$;
3. $f(n) = 3n \log_2 n + 5n + 8$, and $g(n) = n \log_2 n$. ◄

After you have done that show that $f(n) \in O(g(n))$.

► **2.2.** Show that $\log_2 n \in O(\log_{10} n)$. Is it also true that $\log_{10} n \in O(\log_2 n)$? ◄

► **2.3.** Let f, g, h be functions taking on non-negative values. Show the following:

1. If there are constants $C, C' > 0$ such that $f(n) \leq Cg(n)$ for all $n > 0$, and $g(n) \leq C'h(n)$ for all $n > 0$, then there exists a constant $C'' > 0$ such that $f(n) \leq C''h(n)$.
2. If there are constants $C, C' > 0$ such that $f(n) \leq Ch(n)$ and $g(n) \leq C'h(n)$ for all $n > 0$, then there exists a constant $C'' > 0$ such that $f(n) + g(n) \leq C''h(n)$.

In both cases, express C'' in terms of C and C'. ◄

To show that $f(n) \notin O(g(n))$, one way is showing that no matter how we choose C and n_0, there is always some $n > n_0$, such that $f(n) > C \cdot g(n)$. More formally

$$f(n) \notin O(g(n)) \iff \forall C, n_0 > 0, \exists n > n_0 \text{ such that } f(n) > C \cdot g(n).$$

► **2.4.** In each of the following cases, show that $f(n) \notin O(g(n))$.

1. $f(n) = n^2$, and $g(n) = n$;
2. $g(n) = 5n^3$, and $g(n) = n^2$;
3. $f(n) = n \log_2 n$, and $g(n) = n$. ◄

2