EC330 Applied Algorithms and Data Structures for Engineers
Fall 2018

Homework 2

Out: September 14, 2018
Due: September 25, 2018

This homework has a written part and a programming part. Both are due at 8:59 am on September 25. You should submit both parts on Blackboard. For the written part, you should submit a single PDF file containing either typeset answers or scanned copies of hand-written answers. Make sure you write your answers clearly. For the programming part, your code should be easy to read and understand, and demonstrate good code design and style.

1. Asymptotic Comparison [30 pt]

In each of the following situations, indicate whether $f = O(g)$, or $f = \Omega(g)$, or both (i.e. $f = \Theta(g)$). **[2 pt each]**

	$f(n)$	$g(n)$
a.	$n - 100$	$n - 300$
b.	$n^{1/2}$	$n^{2/3}$
c.	$50n + \log n$	$n + (\log n)^2$
d.	$n \log n$	$100n(\log 10n)$
e.	$\log 2n$	$\log 5n$
f.	$10 \log n$	$\log(n^2)$
g.	$n^{1.01}$	$n \log^2 n$
h.	$n^2 / \log n$	$n (\log n)^2$
i.	$(\log n)^{\log n}$	$n / \log n$
j.	$\sqrt{n}$	$(\log n)^3$
k.	$n^{1/2}$	$5^{\log_2 n}$
l.	$n2^n$	3^n
m.	$n!$	2^n
n.	$(\log n)^{\log n}$	$2^{(\log_2 n)^2}$
o.	$\sum_{i=1}^{n} i^k$	n^{k+1}

2. Asymptotic Analysis [10 pt]

Prove that $(\log n)^{10} = O(n^{0.1})$. Show all steps. *Hint: Use L'Hôpital's Rule.*

3. Programming [60 pt]

Make sure to write your name and BU ID in a comment at the top of the program, along with your collaborator's name and BU ID, if any. Your program *must* compile and run on the lab computers. To use the compiler on the lab computers, run "*module load gcc*" first.

a) Consider the infinite sequence 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, …. Write a program that outputs the k^{th} (k>0) digit in this sequence. For example, the 1^{st} digit is 0, the 3^{rd} digit is 2, and the 12^{th} digit is 0. The function declaration is given below.

int kthDigit(int k);

Your job is to implement the *kthDigit* function in a file named "Problem3a.cpp". Submit the single file "Problem3a.cpp" on Blackboard. **[20 pt]**

b) Write a program that accepts an integer array *nums* and returns the sum *closest to 333* by adding up *three* integers in this array. For example, if the *nums* = [20, 120, 200, 5], then the function should return 340 because 340 = 200 + 120 + 20 is closer to 333 than 325 = 200 + 120 + 5. If there is a *tie*, output the *larger* sum. The function declaration is given below.

int sumTo333(vector<int> nums);

Your job is to implement the *sumTo333* function in a file named "Problem3b.cpp". Submit the single file "Problem3b.cpp" on Blackboard. The most efficient solution(s) will receive an extra credit of 0.1 point. **[40 pt]**