Algorithms
Homework 2

Saad Mneimneh
Computer Science
Hunter College of CUNY

Problem 1
Consider the following scenarios, involving unsorted arrays:

- An array A contains n elements, all identical and equal to x, except k of them which are equal to $y \neq x$. Assume $n \geq 2k+1$ and $k \geq 1$. We would like to find the values of x and y in an efficient way, using an oracle who, given two integers a and b can find $\sum_{i=a}^{b} A[i]$ in $O(1)$ time. Describe the most efficient algorithm for this task.

- For the scenario above, find i such that $A[i] = y$. Describe your algorithm and analyze its running time.

- An array A is not sorted but satisfies $|A[i] - A[i+1]| \leq \epsilon$ for all $1 \leq i < n$. Given x such that $A[1] \leq x \leq A[n]$, find an element of A that is within ϵ of x, or determine that no such element exists. What if the condition $A[1] \leq x \leq A[n]$ does not hold?

Problem 2
(Optional for fun, you might want to work on a subset of these)

Rank the functions by order of growth, where $f_1(n) = \Omega(f_2(n)), f_2(n) = \Omega(f_3(n)), \ldots, f_{29}(n) = \Omega(f_{30}(n))$. Use Θ instead of Ω if $f_i(n) = \Theta(f_{i+1}(n))$. Note that $\log^* n$ is the number of times the logarithm function must be applied until the result is less or equal to 1.

n^2	$n^{2 \lg \lg n}$	$n^2 + 2^{100}n$	$\lfloor n \rfloor$	n^n	2^{2n}
$(\frac{3}{2})^n$	$(\frac{2}{3})^n$	$n^{\log_8 n}$	$(\lg n)!$	100^{100}	$(1/n)^{1/\lg n}$
$\ln \ln n$	$2^{\lg^* n}$	$n2^n$	$3(n!)$	$\ln n$	1
$2^{\lg n}$	$(\lg n)^{\lg n}$	e^n	$\sum_{k=1}^{n} k$	$(n+1)!$	$\sqrt{\lg n}$
$\lg(\lg^* n)$	$\lg^*(\lg n)$	n	2^n	$n \lg n$	$2^{2^{n+1}}$

Hint: Sometimes to compare two functions, it helps to compare their logarithms.

Problem 3
Consider the following weird recurrence we saw in class:

$$T(n) = 4T(n/2) + \Theta(n^2/\log n)$$

(a) Show that $T(n) = O(n^2 \log n)$ and that $T(n) = \Omega(n^2)$.

(b) Based on the above, make a guess for $T(n) = \Theta(n^2 \log \log n)$ and prove it using the substitution method.

(c) Using the fact that $1+1/2+1/3+\ldots+1/n = \Theta(\log n)$, use the recursive tree method to show that $T(n) = \Theta(n^2 \log \log n)$.

Problem 4
Give asymptotic upper and lower bounds for $T(n)$ which are as tight as possible. Assume that $T(n)$ is constant for $n \leq n_0$, where n_0 is a constant. For most of these, you can use the Master method. If not, find other ways such as guessing the answer and verifying it using the substitution method.

(a) $T(n) = 6T(n/3) + n^3$

(b) $T(n) = 6T(n/3) + n$

(c) $T(n) = 9T(n/3) + n^2$

(d) $T(n) = 8T(n/2) + n^3 \lg^2 n$

(e) $T(n) = 10T(n/3) + n^2\sqrt{n}$

(f) $T(n) = T(n/3) + 2T(n/4) + n$

(g) $T(n) = T(n^{1/3}) + \lg n$
Hint: change of variable.

(h) $T(n) = 3T(n-1) + n^3$
Hint: Guess from recursive tree and verify, or work directly with a sum.

(i) $T(n) = T(\lg n) + 1$
Hint: Review the definition of $\log^*$, and verify your guess by substitution method.

(j) $T(n) = T(n/4) + \sqrt{n}$

(k) $T(n) = T(n/2 + \sqrt{n}) + 1$
Hint: use the idea of lower and upper bounding this by two recurrences that are more friendly.