COMP 250 2016, Assignment 3 *v3*.0
Due Friday March 18th 2016

1. In all of the following sub-questions, give an algorithm in recursive form and show its running time using the Master Theorem or just by solving the recurrence in the simpler cases.

[35%]

A) Describe a $\Theta(n)$-time algorithm that, given a set S of n integers, determines which two elements in S have the largest sum.

B) Describe a $\Theta(n)$-time algorithm that, given a set S of n integers, determines which two elements in S have the largest difference.

C) Describe a $\Theta(n)$-time algorithm that, given a set S of n integers, determines which two elements in S have the smallest sum.

D) Describe a $\Theta(n \lg n)$-time algorithm that, given a set S of n integers, determines which two elements in S have the smallest difference.

E) Describe a $\Theta(n \lg n)$-time algorithm that, given a set S of n integers and another integer x, determines whether or not there exist two elements in S whose sum is exactly x.

F) Describe an $\Theta(n \lg n)$-time algorithm that, given a set S of n integers and another integer x, determines whether or not there exist two elements in S whose difference is exactly x.

G) Describe an $\Theta(n \lg n)$-time algorithm that, given a set S of n integers and another integer x, determines whether or not there exist two elements in S whose difference is smaller or equal to x.

[30%]

2. On January 19, 2016, Dr. Curtis Cooper published about his recent discovery of a 49th Mersenne prime, $2^{74207281} - 1$ (a number with 22,338,618 digits), as a result of a search executed by a GIMPS server network.

The best method presently known for testing the primality of Mersenne numbers is the Lucas–Lehmer test. Specifically, it can be shown that for prime $p > 2$, $M_p = 2^p - 1$ is prime if and only if

$$S_{p-2} = 0,$$

where $S_0 = 4$ and, for k > 0,

$$S_k = ((S_{k-1})^2 - 2) \ \% \ M_p.$$

I give you two options: you may use either the grade school algorithm for multiplication or the Karatsuba algorithm.

I arbitrarily chose the following two constants:

time to add two numbers of 64 bits each: 0.5 nano-sec
time to multiply two numbers of 64 bits each: 1.5 nano-sec.

How much time would be required (on a single machine) to prove the primality of the 49th Mersenne prime?
(Proceed by making some reasonable assumptions about the recurrence expressing the running-time of both multiplication methods and solving them.)

Count time for subtraction to be the same as addition and the time for modulo (%) to be the same as multiplication.

[35%] 3. Let's consider one last time the problem related to HW-1… but this time I change the conditions slightly:

You receive a list of integers, one at a time, $a_1, a_2, \ldots$ and I want you to find the first time an a_i comes up that has occurred previously, i.e. find the smallest i such that all the $a_1, a_2, \ldots, a_{i-1}$ are distinct but $a_i = a_j$ for some $1 \leq j < i$. However, contrary to HW-1 and HW-2, we do **not** have the nice repetitive structure past position j that $a_{j+k} = a_{i+k}$ for all $k \geq 0$. (This allowed us to obtain an $O(i)$ time algorithm in HW-2.)

Describe an $O(i \log^2 i)$ time algorithm to find the smallest i. (Give an algorithm in recursive form and show its running time using the Master Theorem.)

Note — running time is *not* a function of the global number of elements but only a function of the position of the first repetition.

Hints: Combine merge-sort, binary search and let yourself be inspired by the expansion process of ArrayLists…