الجامعة المصرية اليابانية للعلوم والتكنولوجيا E-JUST Egypt - Japan University of Science and Technology エジプト日本科学技術大学	**Assignment 4: Spring 2021**
Department	Computer Science and Engineering
Submission due	Tue, Apr 27th, 2021 by 9:30am
Date	22/04/2021
Course Title	CSE 326: Analysis and Design of Algorithms
Instructor	Prof. Walid Gomaa
Allowed Equipment	None

Answer all of the following questions

1. Suppose you are to choose between the following three algorithms:
   1. Algorithm A solves the problem by dividing it into five sub-problems each half the size, recursively solving each sub-problem, and then combining the solutions in linear time.
   2. Algorithm B solves the problem of size n by recursively solving two sub-problems of size $n-1$ and then combining the solutions in constant time.
   3. Algorithm C solves the problem of size n by dividing it into 9 sub-problems each of size $n/3$, recursively solving each sub-problem, and then combining the solutions in $\mathcal{O}(N^2)$ time.

   What are the running times of each of these algorithms, and which would you choose?

2. How many lines, as a function of n (in $\Theta(n)$ form), does the following program print? Write a recurrence and solve it. You may assume n is a power of 2.

Algorithm 1 Printing

```
function f(n)
    if n > 1 then
        print_line("Still going")
        f(n/2)
        f(n/2)
    end if
end function
```

3. You are given an infinite array $A[\cdot]$ in which the first n cells (the value of n is unknown) contain integers in sorted order and the rest of the cells are filled with ∞. Describe an algorithm that takes an integer x as input and finds a position in the array containing x, if such a position exists, in $\mathcal{O}(\log n)$ time.

4. Given a sorted array of distinct integers $A[1, \ldots, n]$, you want to find out whether there is an index i for which $A[i] = i$. Give a divide-and-conquer algorithm that runs in time $\mathcal{O}(\log n)$.

5. Consider the task of searching a sorted array $A[1, \ldots, n]$ for a given element x: a task we usually perform by binary search in time $\mathcal{O}(n)$. Show that any algorithm that accesses the array only via comparisons (that is, by asking questions of the form "is $A[i] \leq z$?"), must take $\Omega(\log n)$ steps.

6. Show that any array of integers $A[1, \ldots, n]$ can be sorted in $\mathcal{O}(n+M)$ time, where $M = \max_i x_i - \min_i x_i$. For small M , this is linear time: why does not the $\Omega(n \log n)$ lower bound apply in this case?