CS2800 Design and Analysis of Algorithms
Problem Set #0 January 16, 2023

1. Recall the definition of Big-Oh: We say that $f(n) = O(g(n))$ if $\exists c > 0, n_0 > 0$ such that for every $n \geq n_0$, $f(n) \leq cg(n)$. Show that the definition given above is equivalent to the following definition.

   $f(n) = O(g(n))$ if $\exists c > 0$ such that for every $n \geq 1$, $f(n) \leq cg(n)$.

2. For the following functions $f(n)$ and $g(n)$, answer what is the tightest asymptotic relation between $f(n)$ and $g(n)$.
   (a) $f(n) = n^{\log_2 c}$ and $g(n) = c^{\log_2 n}$ where $c > 2$ is a constant.
   (b) $f(n) = 2^{n^2}$ and $g(n) = n^2 2^{2n}$.
   (c) $f(n) = (\log n)!$ and $g(n) = n^2$.
   (d) $f(n) = n^{\frac{\log\log n}{\log n}}$ and $g(n) = n^{1/10}$.

3. Is $2^{O(n)} = O(2^n)$? If yes, give a proof. Otherwise, give a counter-example.

4. Is it true that for all functions $f(n)$, $f(n) = \Theta(f(n/2))$? If yes, give a proof. Otherwise, give a counter-example.

5. Does there exist an $\epsilon > 0$ such that $\log n = \Omega(n^\epsilon)$?

6. You have devised a new algorithm that you title *ternary search*. The algorithm is as follows:

```
TernarySearch(A, k)
  Input: Sorted array A[1, 2, ..., n] and a key k
  Output: Index of k, if present, else −1
1 if A is empty then return −1
2 if A[n/3] = k then
3     return n/3
4 else
5     if A[2n/3] = k then
6         return 2n/3
7     else
8         if k < A[n/3] then
9             Ternary Search(A[1, 2, ..., n/3 − 1])
10        else
11            if A[n/3] < k < A[2n/3] then
12                Ternary Search(A[n/3 + 1, ..., 2n/3 − 1])
13            else
14                Ternary Search(A[2n/3 + 1, ..., n])
```

   Is your algorithm better than binary search? Why/Why not. Support your reasons mathematically.

7. Recall the *Tower of Hanoi* problem: You are given n disks on a peg (peg 0), and two additional pegs (peg 1 and 2). You goal is to move all the disks from peg 0 to peg 1 or 2 under the following constraint: You can only move one disk at a time, and you are not allowed to place a larger disk above a smaller disk.

   (a) Describe an algorithm to perform the task (this will be recursive). Write a recurrence relation for the number of movements of disks your algorithm makes, and compute the value exactly.

   (b) Consider the following additional constraint to the problem: You are only allowed to move a disk from peg 0 to 1, 1 to 2 or 2 to 0. Design an algorithm to move all the disks from peg 0 to one of the other pegs with this additional constraint. How many movements of disks does your algorithm make now?

8. Smullyan[1] Island has three types of inhabitants: *knights*, who always speak the truth, *knaves* who always lies, and *normals* who speak truth sometimes, and lie sometimes. Everyone knows everyone else's names and type. Your goal is to know everyone's type. You can ask any inhabitant, of another inhabitant's type (this is the only question you are allowed to ask), but you cannot ask his/her own type. Asking the same question multiple times to a person will yield the same answer.

   (a) Suppose that a strict majority of inhabitants are knights, give an efficient algorithm to find the type of every inhabitant.

   (b) Prove that if the number of knights is at most half the total number of inhabitants, then it is impossible to find the type of every inhabitant correctly.

[1] Raymond Smullyan was a mathematician and philosopher famous for his many logic puzzles using self-reference involving knights and knaves. Check out What is the name of this book?