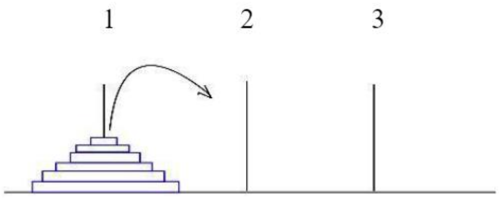
EL9343 Homework 3

Due: Feb.13th 5:00 p.m.

1. True or False questions:
   (a) One can find the maximum sub-array of an array with n elements within $O(n \log n)$ time.
   (b) In the maximum sub-array problem, combining solutions to the sub-problems is more complex than dividing the problem into sub-problems.
   (c) Bubble sort is stable.
   (d) When input size is very large, a divide-and-conquer algorithm is always faster than an iterative algorithm that solves the same problem.
   (e) It takes $O(n)$ time to check if an array of length n is sorted or not.
   (f) Insertion sort is NOT in-place.
   (g) The running time of merge-sort in worst-case is $O(n \log n)$.

2. Consider sorting n numbers stored in array A, indexed from 1 to n. First find the smallest element of A and exchanging it with the element in $A[1]$. Then find the second smallest element of A, and exchange it with $A[2]$. Continue in this manner for the first $n-1$ elements of A.
   (a) Write pseudo-code for this algorithm, which is known as *selection sort*.
   (b) What loop invariant does this algorithm maintain?
   (c) Give the best-case and worst-case running times of selection sort in Θ-notation.

3. The Tower of Hanoi consists of 3 rods and a number of disks of various diameters, which can slide onto any rod. The puzzle begins with **n** disks stacked on a **start** rod in order of decreasing size, the smallest at the top, thus approximating a conical shape. The objective of the puzzle is to move the entire stack to the **end** rod, obeying the following rules:
   i Only one disk may be moved at a time;
   ii Each move consists of taking the top disk from one of the rods and placing it on top of another rod or on an empty rod;
   iii No disk may be placed on top of a disk that is smaller than it.



Please design a MOVE(n, start, end) function to illustrate the minimum number of steps of moving n disks from start rod to the end rod.
You **MUST** use the following functions and format, otherwise you will not get full points of part (a):

```
def PRINT(origin, destination):
        print("Move the top disk from rod", origin, "to rod", destination)

def MOVE(n, start, end): # TODO: you need to design this function
        pass
```

For example, the output of MOVE(2, 1, 3) should be:

```
Move the top disk from rod 1 to rod 2
Move the top disk from rod 1 to rod 3
Move the top disk from rod 2 to rod 3
```

   (a) Fill in the function MOVE(n, start, end) shown above. You can use Python, C/C++ or pseudo-code form, as you want.

(b) Suppose the minimum number of moves of MOVE(n, 1, 3) is $f(n)$. What's the relationship between $f(n)$ and $f(n-1)$? Please write the recurrence and then show $f(n) = 2^n - 1$.

4. Consider a variation of the Towers of Hanoi problem mentioned in Q3, in which there are **4** rods, instead of 3. All the other rules are the same.
Design a strategy to move the n disks from the first rod to the last rod with at most $2^{O(\sqrt{n})}$ moves, that is, the number of moves should be bounded by $2^{c\sqrt{n}}$ for some constant c. You are free to use the conclusion in Q3: there is a strategy to solve the 3-rod version problem in $f(n) = 2^n - 1$ moves.

   (a) Suppose in the 4-rod version, to move n disks to another rod requires $T(n)$ moves, and we are to show that $T(n) \leq 2^{c\sqrt{n}}$ for some constant c. Now consider we have x disks. First, we move $x - a$ disks to the third rod, then move the remaining a disks to the last rod, and finally move the $x - a$ disks from the third rod to the last rod. Following this way, how can you express $T(x)$? You are supposed to use $T(\cdot)$, x and a in the expression.

   (b) Now we can generate a sequence of recurrences by letting $a = \sqrt{n}$ (suppose n is some k^2 to avoid corner cases), and $x = n, n-a, n-2a, \ldots$. Please solve for the $T(n)$ and show that $c = 2$ is sufficient for $T(n) \leq 2^{c\sqrt{n}}$.