Algorithms Cs545 — Homework #1

1. Let $f_i(n)$ be a sequence of functions, such that for every i, $f_i(n) = O(n)$. Let $g(n) = \sum_1^n f_i(n)$. Prove or disprove: $g(n) = O(n^2)$.

   Answer: *Not true. Take* $f_i(n) = i \cdot n$. *Then*

   $$g(n) = \sum_1^n in = n \sum_1^n i = \Theta(n^3)$$

2. If $f_1(n) = O(g_1(n))$ and $f_2(n) = O(g_2(n))$. Prove or disprove:

   - $f_1(n) + f_2(n) = O(g_1(n) + g_2(n))$

     Answer: *True. We know that there exists positive constnts* n_1, K_1 *(resp.* n_2, K_2*) such that for* $n > n_1$ *(resp.* $n > n_2$*) we have that* $f_1(n) \leq K_1 g_1(n)$ *(resp.* $f_2(n) \leq K_2 g_2(n)$*). Hence for every* $n > \max\{n_1, n_2\}$*, we have that* $f_1(n) + f_2(n) \leq \max\{K_1, K_2\}(g_1(n) + g_2(n))$.

   - $f_1(n) * f_2(n) = O(g_1(n) * g_2(n))$

     Answer: *True. Analogous to the previous case*

   - $f_1(n)^{f_2(n)} = O(g_1(n)^{g_2(n)})$

     Answer: *Not true. take* $f_1(n) = 2, f_2(n) = n,\ g_1(n) = 0.5, g_2(n) = n.$

3. Let $P = \{p_1 \dots p_n\}$ be a set of n distinct points in the plane. Describe an $O(n \log n)$-time algorithm that finds the triangle with smallest perimeter, whose vertices are three different points of P.

4. You are given two arrays A and B, each contains n numbers, and each is sorted in increasing order. Let S denote the set of all numbers which are either in A, in B or in both. Find in time $O(\log n)$ the median of S. (if you have problem finding this element in $O(\log n)$, find it in time $O(\log^2 n)$.

5. Suggest a data structure that supports grades for a student. The operations on the data structure are as follows:

   `Insert`(g, d) — Insert the grade g that the student received, for an example that took place at a date d. Each grade is a number between 1 and 100. For example, `insert`(73,"9/16/02").

`Average`(d_1, d_2) report what is the average of all grades that the student received in exams that took place between date d_1 and date d_2.

Each operation should take time $O(\log n)$, where n is the number of grades store in the sata structure.

Answer: *Store the grades in a standard search tree, where each node μ in the tree stores the total sum s_μ of grades and the number n_μ of grades in the subtree rooted at μ. Then when the query* `Average`(d_1, d_2) *is submitted, we sum (in $O(\log n)$ time) the sum of grades and number of grades between d_1 and d_2. To sum of example the number of grades, it is easy to sum the sum of all grades that occur before d_1 the sum of all grades that occur before d_2 and subtract.*

To find the sum op all grades that occur before d_2, perform `find`(d_2) *in the tree, and trace the path π that the search for d_1 performed in the tree. For every node μ at the tree at which the path turned to the right subtree of a node μ sum the value of $s_{left(\mu)}$, plus the value stored at μ itself. Clealry it is doable in $O(\log n)$*

6. Assume that each point on the interval $[0, 1]$ could be colored either black or white, and that initially the whole interval is white. We define the operation of *reversing* the color of a point $x \in [0, 1]$, as follow: If x is black before the operation, then its color is white after the operation, and vice versa. Suggest a data structure that support the following operations.

   `reverse`(x_1, x_2) — reverse the color of each point of in the interval (x_1, x_2), where $0 < x_1 < x_2 < 1$.

   `report_color`(x) report the color of x, where $x \in [0, 1]$.

   The running time of each operation should be $O(\log n)$, where n here is the number of reverse operations.

   Answer: *Each* `reverse`(x_1, x_2) *command defined an interval, where x_1 is its left endpoint and x_2 is its right endpoint. Observe that a point x is white (resp. black) iff the different between the number of left endpoints to the left of x, and the number of right endponits to the left of x is even (resp. odd). From here, the answer is similar to the previous answer.*