CSCI5010 Exercise List 4

Remark: For the correctness proof of our algorithm that divides a non-monotone polygon into a monotone one, read Chapter 3.2 of the textbook.

Problem 1 (Branch Sizes in a Tree). Let T be a tree of n nodes, where each node has at most 3 neighbors. Let u be a node in T, and e_1, e_2, e_3 be the 3 edges of T adjacent to u. For each edge e_i ($i = 1, 2, 3$), we define a *branch size* of u as follows. Imagine removing e_i from T, which will break T into two components; then, the branch size of u with respect to e_i is the number of nodes in the component that does not include u. Give an algorithm to compute all the branch sizes of all the nodes in T. Your algorithm should run in $O(n)$ time.

Problem 2 (Balanced Partitioning of a Tree). Let T be a tree of n nodes (n is a sufficiently large integer), where each node has at most 3 neighbors. Prove that we can always remove an edge of T such that each of the two resulting components has between $(n-1)/3$ and $1 + (2n/3)$ nodes. Give an algorithm to find such an edge in $O(n)$ time.

Problem 3 (Balanced Partitioning of a Polygon). Let P be a polygon (not necessarily monotone or convex) of n vertices where n is a sufficiently large integer. Give an algorithm that adds a diaganol to break P into two polygons, each of which has at least $n/4$ and at most $3n/4$ vertices. Your algorithm should run in $O(n \log n)$ time (hint: use the dual graph of a triangulation).

Problem 4* (Stabbing Number, Exercise 3.13 of the Textbook). The *stabbing number* of a triangulated polygon P is the maximum number of diagonals that can be intersected by any line segment inside P. Give an algorithm that computes a triangulation of a *convex* P that has a stabbing number of $O(\log n)$.