Triangulating a Monotone Polygon

1. Problem Statement
   - triangulating y-monotone polygons
2. An Incremental Triangulation Algorithm
   - walking left and right boundary chains
   - pseudo code for the algorithm
3. Linear Time
   - cost analysis

pseudo code – the initialization and loop

Algorithm TRIANGULATEMONOTONEPOLYGON(P)

Input: a doubly connected edge list $\mathcal{D}$
stores a strictly y-monotone polygon P.
Output: the updated $\mathcal{D}$ stores a triangulation of P.

1. Merge vertices of the left and right chains in $[u_1, u_2, \ldots, u_n]$, sorted on their y-coordinate, leftmost breaks ties, in descending order.
2. Initialize the stack S, push u_1 and u_2 onto S.
3. For j from 3 to $n - 1$ do
4. process vertex u_j.

The statement "process vertex u_j" is explained in the next two slides.

processing vertices on opposite chains

3. For j from 3 to $n - 1$ do
4.   if u_j and Top(S) are on opposite chains then
5.   for all $u \in S \setminus$ Bottom(S) do
6. $u = \text{pop}(S)$
7.   insert diagonal (u_j, u) into $\mathcal{D}$
8. $u = \text{pop}(S)$
9.   push(S, u_{j-1}); push(S, u_j)
10.   else ...

The popping of all vertices and the removal of Bottom(S) corresponds to triangles splitting off.

Exercise 1: Explain why the diagonals (u_j, u) are inside P. In your proof, take into account that P is y-monotone and the processing order of the vertices.

processing vertices on the same chain

10. else
11. $u_\ell = \text{pop}(S)$
12. $u = u_\ell$
13. while the diagonal $(u_j, u) \in P$ do
14. insert (u_j, u) into $\mathcal{D}$
15. $u = \text{pop}(S)$
16. push(S, u_ℓ); push(S, u_j);
17. Add diagonal from u_n to all $u \in S$ except for Top(S) and Bottom(S).

Exercise 2: Using your solution to Exercise 1 as a Lemma, prove the correctness of Algorithm TRIANGULATEMONOTONEPOLYGON.