The two exercises below can be solved independently and in any order. All arguments should be expressed in a rigorous and clear manner.

**Exercise 1 – Efficiently algorithms in planar graphs**

In this exercise we consider simple planar graphs\(^1\) (no loops, no multiple edges) and we address the problem of efficiently and listing the 4-cliques (complete sub-graphs of size 4). A triangle is a cycle consisting of 3 distinct vertices (equivalently, a triangle is a 3-clique, a complete graph on 3 vertices): observe that a triangle do not necessarily defines faces in the planar embedding of a graph (refer to Fig. 1). A stack triangulation \(T\) is a plane triangulation\(^2\) defined as follows (see Fig. 1 for an illustration): \(T\) is just one triangle, or \(T\) can be obtained from a stack triangulation subdividing a face \(p\overline{q}\overline{r}\) into three faces \(p\overline{q}\overline{t}\), \(q\overline{r}\overline{t}\), \(p\overline{r}\overline{t}\).

Figure 1: Examples of stack triangulations. The triangle \(\{b, c, d\}\) defines a face in the embedding of \(T\) but it does not define a face in the embeddings of \(T'\) and \(T''\). The sets of vertices \(\{a, c, d, f\}\) and \(\{a, d, f, g\}\) define two 4-cliques in \(T''\).

### Counting triangles in planar graphs

**Question 1.1** Give a linear bound on the number triangles that a stack triangulation of \(n\) vertices can have.

\(^1\)In this exercise you may assume that you are provided with a representation of the embedding of the input graph \(G\) (e.g. the half-edge representation) and with a data structure for efficiently testing in \(O(1)\) time whether an edge \((u, v)\) belongs to \(G\) (using for example an adjacency matrix or a Hashing table storing the pairs \(\{u, v\}\)).

\(^2\)Recall that a triangulation is a simple (cellularly embedded) planar graph where all faces have degree 3.
Question 1.2 Let $G$ a simple planar graph with $n$ vertices. Give a bound on the number of triangles that $G$ can contain.

**Listing all 4-cliques in linear time**

Let us consider a partition of the vertices of $G$ into $k + 1$ sets $V_0, V_1, V_2, \ldots, V_k$ obtained by computing a BFS tree (according to a breadth-first search) whose root is an arbitrary vertex $r$. By definition $V_j$ is the set of vertices at distance $j$ from the root $r$ (so $V_0 = \{r\}$). Let us denote by $E_j$ the set of edges $e = (u, v)$ such that $u \in V_{j-1}$ and $v \in V_j$ (an edge belongs to $E_j$ if it is connecting two vertices on levels $V_j$ and $V_{j-1}$).

Question 1.3 Consider a 4-clique $Q = \{u, v, w, x\}$ in $G$. Show that the four vertices $u, v, w, x$ cannot all belong to the same level $V_j$.

Question 1.4 Consider a 4-clique $Q = \{u, v, w, x\}$ in $G$, and let $j$ be a positive integer $\leq k$.

- assume $u \in V_{j-1}$ and $v, w, x \in V_j$. Show that for one of the tree vertices $v, w, x$ the only incident edge lying in $E_j$ has $u$ has other extremity.
- assume $u, w, x \in V_{j-1}$ and $x \in V_j$. Show that the edges incident to $x$ lying in $E_j$ are exactly $(u, x), (v, x)$ and $(w, x)$.
- assume $u, v \in V_{j-1}$ and $w, x \in V_j$. Show that one of the vertices $w, x$ has exactly two incident edges lying in $E_j$ (whose other extremities are $u$ and $v$).

Question 1.5 Based on the case analysis of previous question, devise a linear time algorithm $\text{enumerate}(G, L)$ that allows us to list all 4-cliques of the input planar graph $G$, provided with the complete list $L$ of all triangles contained in $G$ (which is assumed to be pre-computed).

**Remark:** You are asked to provide a high level description, as well as the pseudo-code, of your algorithm and to justify its runtime complexity (with respect to the parameter $n$, the size of the input planar graph).

Exercise 2 – Triangulations with boundaries

Let us consider a plane quasi-triangulation $T$: a simple planar graph whose inner faces have all degree 3, and where there is a single face of arbitrary degree, called the outer face (we say that the number of boundaries is $b = 1$).

Let us denote by $v_i \ (i \geq 3)$ the number of inner vertices of degree $i$ of $T$ (the degree of a vertex is the number of its neighbors). Let $b_j \ (j \geq 2)$ be the number of boundary vertices (incident to the outer face) having degree $j$.

**Question 2.1** Show that the vertex degrees satisfy the following relation:

$$\sum_{i \geq 3} (6 - i) v_i + \sum_{j \geq 2} (4 - j) b_j = 6$$

What happens when there are several boundaries ($b > 1$)?