The program is written in C++ and I use Microsoft Visual Studio 2013 for Windows as IDE and compiler.
However it should be quite easy to change it to use another compiler.

The first class is the **Graph** class in the Data folder.
It's basically an array of nodes and an array of edges.
The **Edge** class contains the zero-based indices of the 2 nodes, as well as the number of triangles containing the edge.
The **Node** class stores the zero-based indices of the edges containing this node.

Then an important class is the **GraphGenerator** class: as input it takes a graph with *n-1* edges and it generates all graphs with *n* edges containing this graph.

The function **Graph::CanAddEdge** does the following checks:

- If there is already an edge connecting the 2 nodes, it returns false (two nodes can't be connected by more than one edge).
- If the nodes to connect are the 2 opposite nodes of 2 triangles sharing a same edge (this is the first rule presented above), it returns false.
- If adding the edge would create one or two new triangles and this would lead to an edge shared by more than 2 triangles, it returns false (this is the third rule presented above).

The second important class is the **IsomorphismDetector** class: it takes two graphs and says if they are isomorphic.
It tries to map a node of the first graph to a node of the second graph which has the same number of edges. Then the algorithm is recursive until all nodes have been mapped (or not).

The last class is **GraphValidator** which detects some invalid cases: the one associated to the rule # 2, the 3 invalid graphs with 9 edges and the invalid graph with 10 edges.