Fleury’s Algorithm

Input: Adjacency list ZA of a graph $G = (V, E)$ without vertices of odd degree.

Output: Euler cycle represented as a sequence of vertices stored in stack CE

```
begin
     STACK := $\emptyset$;
     CE := $\emptyset$
     v := any vertex of the graph $G$;
     STACK.PUSH(v) // inserts v at the top of the stack STACK
     while STACK $\neq \emptyset$ do
          begin
               v := top(STACK); // reads the top of the stack STACK (without deleting it from the STACK)
               if ZA[v] $\neq \emptyset$
                    begin
                         u := the first vertex from the list ZA[v];
                         STACK.PUSH(u); // inserts u at the top of the STACK
                         ZA[v] := ZA[v] \ v; ZA[u] := ZA[u] \\ v // deletes v form the graph G
                         v := u;
                    end
               else // the list ZA[v] is empty
                    begin
                         v := STACK.POP; // deletes and returns the object from the top of the stack Stack
                         CE.PUSH(v) // inserts v at the top of the stack CE
                    end
          end
end
```