Project on Eulerian graphs and the Fleury Algorithm for cycle detection.

Authors: BOUSSAD Mohamed & GUILLAUME Quentin

Language of choice: Java. Requires JRE 16 and above (uses the `var` keyword and `try-with-resources`)

Building and running

This project is built using Gradle. Build it using `./gradlew build`

If you're using **Intellij IDEA**: Run it directly from your IDE. Change the program arguments to supply different graph files.

Otherwise, you should find the built JAR in the `build/libs` directory. Run it using `java -jar <JAR> <graph_file>`, where `<JAR>` is the path to the built JAR file and `<graph_file>` is the path to the graph to test.

Time & Space Complexity

V represents the amount of vertices in the graph.
E represents the amount edges in the graph.

The main class implements three core methods:

Method	Time Complexity	Space Complexity
`GraphUtils.cloneGraph`	O(V + E)	O(V + E)
`GraphUtils.removeEdge`	O(V)	O(V)
`GraphUtils.edgeExists`	O(d(u))	O(1)
`GraphUtils.countTotalEdges`	O(V + E)	O(1)
`fleury`	O(E * (V + E))	O(V + E)
`isEulerian`	O(V + E)	O(V)
`isConnected`	O(V + E)	O(V)
`isIsthme`	O(V + E)	O(V + E)
`isCycleEulerian`	O(V + E)	O(E)