Due: 18.01.2026 Sunday 23:59

EE-471 Power System Analysis I
2025-2026 Fall Term Project

Aim: To implement a Python function for Fast-Decoupled Load Flow (FDLF) analysis using Python 3.13 or newer. The function must be generic, meaning it should be capable of solving the load flow for different power systems, provided they are described using the specified input format.

Inputs: Your function should take paths to two input files.

1. **Topology File:** Includes connectivity and parameters of the power system network. The file will be named [case_name].json, where [case_name] is the name of the test case. An accompanying markdown file will explain the data format.
2. **Load Flow Case File:** Includes the load and generation data as well as the load flow bus types. The file will be named [case_name]_load_flow.json where the [case_name] is the name of the test case. An accompanying markdown file will explain the data format.

Outputs: Your function should output 4 variables.

1. Voltage magnitude values at each bus in per unit (voltage_magnitude)
   Explanation: Dictionary where keys are bus IDs and values are voltage magnitudes in per unit.

2. Voltage angle values at each bus in radians (voltage_angle)
   Explanation: Dictionary where keys are bus IDs and values are voltage angles in radians.

3. Total active power losses in the system (p_loss)
   Explanation: Float in MW

4. Total reactive power losses in the system (q_loss)
   Explanation: Float in MVAR

Submission: Submit the final version of your code via GitHub classroom and the final report should be submitted via ODTUClass. Do **NOT** include detailed explanations of your code in your report. The report should include the following items:

a) Results of load flow analysis. Compare your results with Newton-Raphson PF using PSSE (for all cases) and DigSilent (for the 14-bus and 30-bus cases). You are expected to explain any including convergence issues. Include graphs and tables.
b) An analysis of how the solution time and error change with the convergence threshold.
c) A description of any methods used to improve computational performance.
d) Flowcharts of your implemented code.
e) Explanation of the solution.
f) Any additional reasoning and comments.
g) A plot of the YBUS matrix sparsity pattern.

Bonus points will be awarded for computationally efficient solutions provided that the execution time will be comparable with that of state-of-the-art solvers. Note that to be eligible for bonus evaluation your project should also be demonstrably correct across all test cases.

Note 1: You are responsible for ensuring your project runs correctly. Example results will not be provided. Use PSSE and DigSilent to test and validate your project.

Note 2: Follow the variable name conventions and order for outputs.