Unification Algorithm Practical Question

A company is developing an **AI-powered chatbot** that helps students answer queries about their courses.
The chatbot uses **first-order predicate logic** to reason about knowledge, such as:

- Teaches(Professor, Subject)
- Studies(Student, Subject)
- Advises(Professor, Student)

To answer queries like *"Who teaches John's subject?"*, the chatbot needs to **unify logical expressions** to match facts with rules.

Scenario:

The knowledge base has the following rules:

1. Teaches(DrSmith, AI)
2. Studies(John, AI)
3. Advises(x, y) ← Teaches(x, z) ∧ Studies(y, z)

The chatbot receives a query:
"Who advises John?"

To solve this, it must unify the query with the rule Advises(x, y) and then check substitutions from the knowledge base.

QUESTIONS:

1. Implement a Python program using the **unification algorithm** that can unify two predicate expressions (e.g., Advises(x, John) with Advises(DrSmith, John)).
2. Use your implementation to show how the query *"Who advises John?"* can be answered by applying unification on the given knowledge base.
3. Show the final substitution result.