Lecture Notes No. 10			
Topic:	**Rule-Based Expert Systems**	**Week No.**	15
Course Code:	**CSST101**	**Term:**	1st Semester
Course Title:	**Advance Knowledge Representation and Reasoning**	**Academic Year:**	2025-2026
Student Name		**Section**	
Due date		**Points**	

Learning Outcomes

By the end of this lesson, you should be able to:

1. Explain the concepts and components of rule-based expert systems.
2. Represent domain knowledge using IF–THEN rules.
3. Implement a prototype expert system in Python.
4. Collaborate to define a domain and knowledge base for reasoning tasks.

1. What is a Rule-Based Expert System?

A rule-based expert system is a type of AI that **encodes expert knowledge as rules** and uses an inference engine to make decisions or solve problems.

Definition:

- **Knowledge Base:** Collection of facts and rules about a domain.
- **Inference Engine:** Applies rules to infer new knowledge.
- **Working Memory:** Stores current facts and intermediate conclusions.

Example:

- Domain: Medical diagnosis
- Rule: "IF patient has fever AND cough THEN possible flu"
- System can ask questions and infer diagnoses based on user input.

2. Key Concepts

Component	Description	Example
Rule	IF–THEN statement encoding knowledge	IF fever AND cough THEN flu
Fact	Statement considered true in the current domain	patient_has(fever)
Inference Engine	Applies rules to derive conclusions	Forward or backward chaining
Forward Chaining	Starts with known facts, applies rules to find new facts	Diagnosis from symptoms

Component	Description	Example
Backward Chaining	Starts with goal, searches rules to validate it	“Does the patient have flu?”

Key Idea:

- Rule-based systems **simulate expert decision-making**.
- Forward chaining is **data-driven**, backward chaining is **goal-driven**.

3. Implementing in Python

Example: Simple rule-based system using IF–THEN logic

```
facts = {"fever": True, "cough": True}
rules = [
    {"if": ["fever", "cough"], "then": "flu"},
    {"if": ["fever", "rash"], "then": "measles"}
]

def forward_chaining(facts, rules):
    conclusions = []
    for rule in rules:
        if all(facts.get(f, False) for f in rule["if"]):
            conclusions.append(rule["then"])
    return conclusions

print(forward_chaining(facts, rules))  # Output: ['flu']
```

Exercise:

- Add more rules for other diseases.
- Test with different combinations of facts.

4. Lab Activity

Goal: Prototype a small rule-based expert system for a chosen domain.

Sample Tasks:

- Define domain facts and rules.
- Implement forward or backward chaining in Python.
- Test the system with sample queries.

Group Activity:

- Define a domain (e.g., smart home, medical, traffic).
- Collaboratively create knowledge base rules for the system.

5. Applications in AI

Field	Example of Use
Medical Diagnosis	Suggest diseases based on symptoms
Smart Homes	Automate devices using rules
Industrial Systems	Fault detection and troubleshooting
Customer Support	Provide automated solutions based on queries

6. Reflection and Discussion

1. What are the advantages and limitations of rule-based systems?
2. How does forward chaining differ from backward chaining?
3. Can you think of a real-world problem suitable for a rule-based system?

7. Summary

- Rule-based expert systems **encode knowledge as IF–THEN rules**.
- The inference engine applies rules to **derive conclusions**.
- Python can simulate expert reasoning for small domains.
- Collaborative rule creation helps define comprehensive knowledge bases.

Self-Check

1. What are the main components of a rule-based expert system?
2. How does forward chaining work?
3. How does backward chaining differ from forward chaining?
4. Give an example of a domain where a rule-based system can be applied.