**Department of Computer Engineering**
**Academic Term II: 23-24**

**Class: B.E (Computer), Sem – VI** **Subject Name: Artificial Intelligence**

**Student Name: Jatin Kadu** **Roll No: 9548**

| | |
|---|---|
| **Practical No:** | **9** |
| **Title:** | Simple Prototype for expert system |
| **Date of Performance:** | **1/4/24** |
| **Date of Submission:** | **6/4/24** |

## Rubrics for Evaluation:

| Sr. No | Performance Indicator | Excellent | Good | Below Average | Marks |
|---|---|---|---|---|---|
| 1 | On time Completion & Submission (01) | 01 (On Time) | NA | 00 (Not on Time) | |
| 2 | Logic/Algorithm Complexity analysis (03) | 03(Correct) | 02(Partial) | 01 (Tried) | |
| 3 | Coding Standards (03): Comments/indention/Naming conventions<br>Test Cases /Output | 03(All used) | 02 (Partial) | 01 (rarely followed) | |
| 4 | Post Lab Assignment (03) | 03(done well) | 2 (Partially Correct) | 1(submitted) | |
| **Total** | | | | | |

**Signature of the Teacher:**

# Experiment No: 9

**Title:** Simple prototype for Expert System

**Objective:** Understanding the composition of expert system for a given prototype

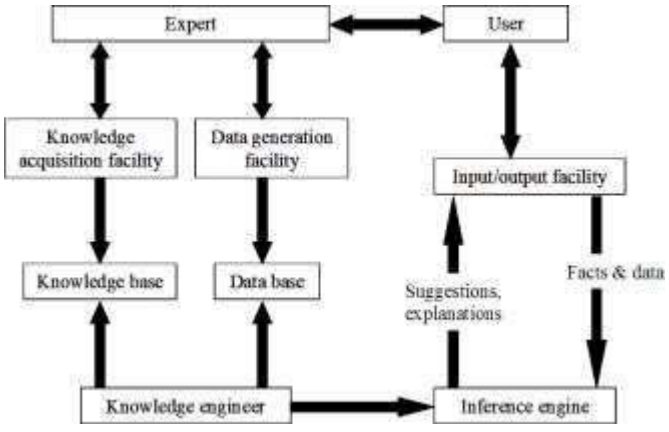
## Theory:

Expert systems are the computer applications developed to solve complex problems in a particular domain, at the level of extra-ordinary human intelligence and expertise. They are one of the prominent research domains of AI. It is introduced by the researchers at Stanford University, Computer Science Department.

### General Characteristics of Expert Systems:

- High performance
- Understandable
- Reliable
- Highly responsive

### General Components of Expert Systems




**General components of Expert System**

**Knowledge Base:** It contains domain-specific and high-quality knowledge. Knowledge is

required to exhibit intelligence. The success of any ES majorly depends upon the collection of highly accurate and precise knowledge.

**Components of Knowledge Base**

The knowledge base of an ES is a store of both factual and heuristic knowledge.

1. **Factual Knowledge** − It is the information widely accepted by the Knowledge Engineers and scholars in the task domain.
2. **Heuristic Knowledge** − It is about practice, accurate judgment, one's ability of evaluation, and guessing.

**Knowledge representation:** It is the method used to organize and formalize the knowledge in the knowledge base. It is in the form of IF-THEN-ELSE rules.

Knowledge Acquisition: The success of any expert system majorly depends on the quality, completeness, and accuracy of the information stored in the knowledge base.

The knowledge base is formed by readings from various experts, scholars, and the Knowledge Engineers. The knowledge engineer is a person with the qualities of empathy, quick learning, and case analyzing skills.

He acquires information from subject experts by recording, interviewing, and observing him at work, etc. He then categorizes and organizes the information in a meaningful way, in the form of IF-THEN-ELSE rules, to be used by an interference machine. The knowledge engineer also monitors the development of the ES.

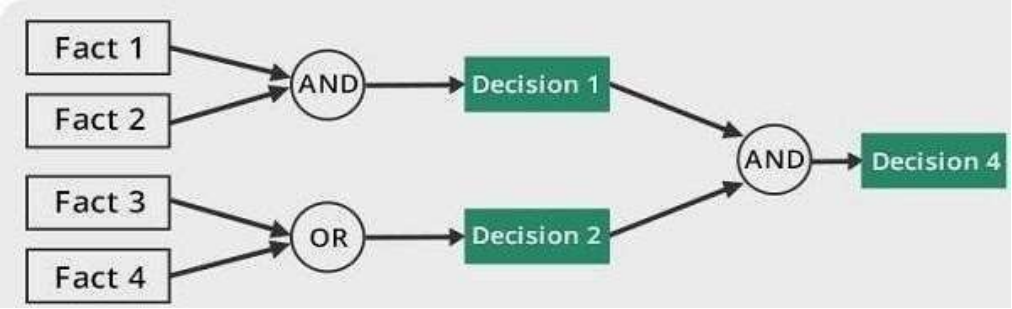
**General Inference Engine:** Use of efficient procedures and rules by the Inference Engine is essential in deducting a correct, flawless solution. In case of knowledge-based ES, the Inference Engine acquires and manipulates the knowledge from the knowledge base to arrive at a particular solution.

In case of rule-based ES, it

- Applies rules repeatedly to the facts, which are obtained from earlier rule application.
- Adds new knowledge into the knowledge base if required.
- Resolves rules conflict when multiple rules are applicable to a particular case.

To recommend a solution, the Inference Engine uses the following strategies –

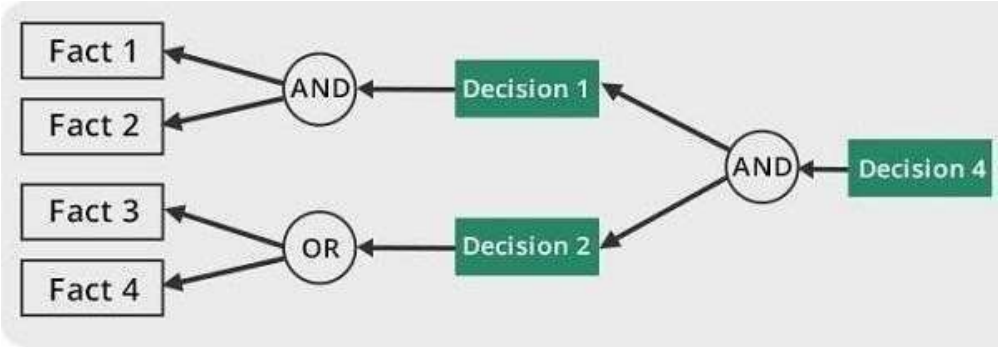
**Forward Chaining:** It is a strategy of an expert system to answer the question, "**What can happen next**?"




Forward Chaining in ID3

Here, the Inference Engine follows the chain of conditions and derivations and finally deduces the outcome. It considers all the facts and rules, and sorts them before concluding to a solution. This strategy is followed for working on conclusion, result, or effect. For example, prediction of share market status as an effect of changes in interest rates.

**Backward Chaining:** With this strategy, an expert system finds out the answer to the question, "**Why did this happen**?"




Backward Chaining in ID3

On the basis of what has already happened, the Inference Engine tries to find out which conditions could have happened in the past for this result. This strategy is followed for finding out cause or reason. For example: diagnosis of blood cancer in humans.

This concludes the inference engine part. The next component is the User interface.

**User Interface:** User interface provides interaction between users of the ES and the ES itself. It is generally Natural Language Processing so as to be used by the user who is well-versed in the task domain. The user of the ES need not necessarily be an expert in Artificial Intelligence.

It explains how the ES has arrived at a particular recommendation. The explanation may appear in the following forms:

- Natural language displayed on screen.
- Verbal narrations in natural language.
- Listing of rule numbers displayed on the screen.

**Requirements of Efficient ES User Interface**

- It should help users to accomplish their goals in shortest possible way.
- It should be designed to work for user's existing or desired work practices.
- Its technology should be adaptable to user's requirements; not the other way round.

**Benefits of Expert Systems**

- **Availability** − they are easily available due to mass production of software.
- **Less Production Cost** − Production cost is reasonable. This makes them affordable.
- **Speed** − they offer great speed. They reduce the amount of work an individual puts in.
- **Less Error Rate** − Error rate is low as compared to human errors.
- **Reducing Risk** − they can work in the environment dangerous to humans.
- **Steady response** − they work steadily without getting emotional, tensed or fatigued

**Expert Systems Limitations**

No technology can offer an easy and complete solution. Large systems are costly; require significant development time, and computer resources. ES's have their limitations which include

- Limitations of the technology
- Difficult knowledge acquisition
- ES are difficult to maintain
- High development costs

**OUTPUT:**

**Post Lab Questions:**

Q1] What are the applications of expert systems?

**Code:**

```
class ExpertSystem:
    def __init__(self):
        self.knowledge_base = {
            'Yellow leaves': 'Nitrogen deficiency',
            'Brown spots on leaves': 'Fungal infection',
            'Wilting leaves': 'Watering issues',
            'White powdery substance on leaves': 'Powdery mildew'
        }

    def diagnose(self, symptoms):
        possible_diseases = []
        for symptom, disease in self.knowledge_base.items():
            if symptom in symptoms:
                possible_diseases.append(disease)
        return possible_diseases

class UserInterface:
    def __init__(self):
        self.expert_system = ExpertSystem()

    def start(self):
        print("Welcome to the Plant Disease Diagnosis System!")
        while True:
            print("\nEnter the symptoms separated by commas (e.g., Yellow leaves, Wilting
leaves):")
            user_input = input("Symptoms: ")
            symptoms = [s.strip() for s in user_input.split(',')]
            diagnoses = self.expert_system.diagnose(symptoms)
            if diagnoses:
                print("\nPossible diseases:")
                for disease in diagnoses:
                    print(f"- {disease}")
            else:
                print("\nNo diagnosis could be made based on the symptoms provided.")
```

```
        choice = input("\nDo you want to diagnose another set of symptoms? (yes/no):
").lower()
        if choice != 'yes':
            print("Thank you for using the Plant Disease Diagnosis System!")
            break

# Example usage:
def main():
    ui = UserInterface()
    ui.start()

if __name__ == "__main__":
    main()
```

**Output:**