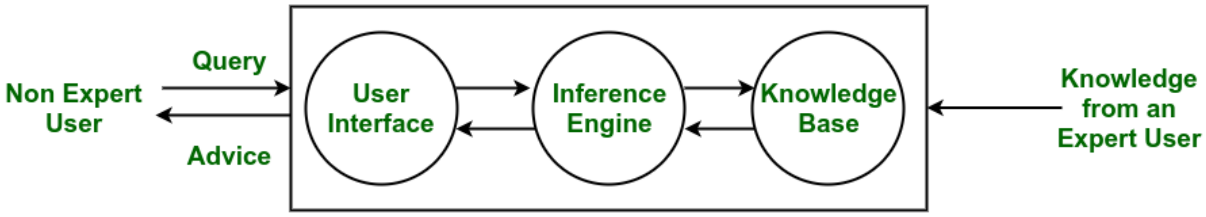
Experiment No: 9 Date:

Expert System

Aim: To implement an Expert System for 4-wheeler Automobile maintenance.

Theory:

An expert system is a type of artificial intelligence (AI) system that emulates the decision-making ability of a human expert in a specific domain or field. It aims to solve complex problems by reasoning through knowledge, heuristics, and inference rules rather than by following predefined algorithms.



An expert system for automobile maintenance aims to provide diagnostic and troubleshooting assistance for common issues that car owners might encounter with their vehicles. This system utilizes a knowledge base of symptoms, potential causes, and recommended actions to guide users in identifying and addressing problems with their cars. Here's how we can implement it:

Components of the Expert System:

- Knowledge Base: This comprises a database of symptoms, potential issues, and corresponding maintenance actions. It forms the foundation of the expert system and is used for diagnosis and recommendations.

- User Interface: The system interacts with users through a user-friendly interface where users can input symptoms and receive guidance on troubleshooting and maintenance.

- Inference Engine: The inference engine is responsible for processing user inputs, matching symptoms to potential issues, and generating recommendations based on predefined rules and logic.

Characteristics of an Expert System :

- Human experts are perishable, but an expert system is permanent.
- It helps to distribute the expertise of a human.
- One expert system may contain knowledge from more than one human experts thus making the solutions more efficient.
- It decreases the cost of consulting an expert for various domains such as medical diagnosis.
- They use a knowledge base and inference engine.
- Expert systems can solve complex problems by deducing new facts through existing facts of knowledge, represented mostly as if-then rules rather than through conventional procedural code.

Advantages :

- Low accessibility cost.
- Fast response.
- Not affected by emotions, unlike humans.
- Low error rate.

Disadvantages :

- The expert system has no emotions.
- It is developed for a specific domain.
- It needs to be updated manually. It does not learn itself.

Algorithm:

```
Algorithm ExpertSystem:
   Input: Knowledge Base (KB), User Query (Q)
   Output: System Response (R)

   1. Acquire knowledge from experts or existing sources
   2. Represent knowledge in a format understandable by the computer
   3. Repeat until termination condition is met:
      a. Receive user query Q
      b. Apply inference engine to infer conclusions based on KB and Q
      c. Provide explanations for the conclusions
      d. Generate system response R
   4. End Algorithm
```

Example:

User Input:
Symptoms: Strange engine noises, rough idling

Diagnosis:
Based on the symptoms provided, the system identifies potential issues such as engine problems and fuel system issues.

Recommendations:
Check engine oil level and quality
Inspect belts and pulleys
Clean or replace spark plugs
Check air filter

Program:

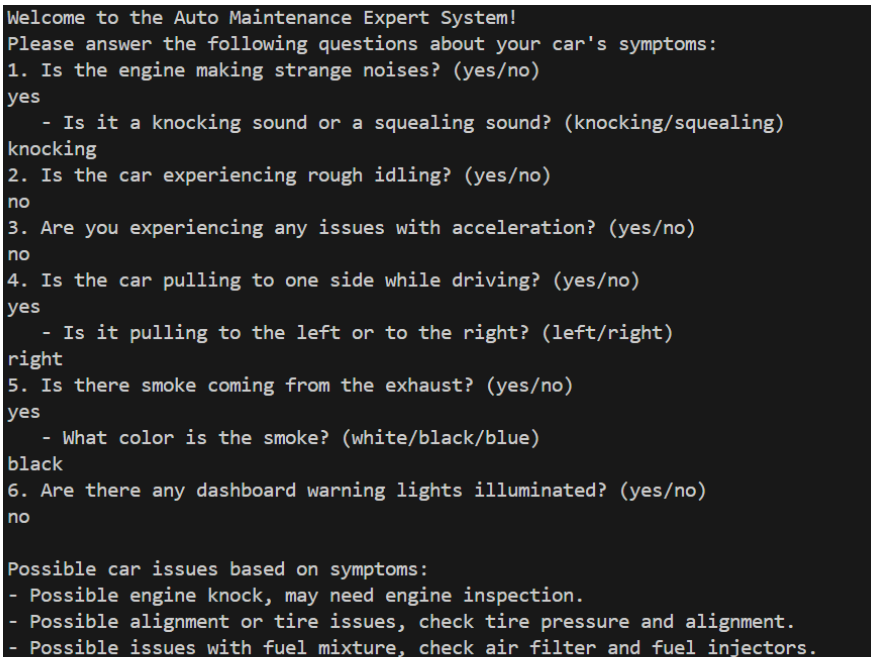
```
def ask_symptoms():
    symptoms = []
    print("Welcome to the Auto Maintenance Expert System!")
    print("Please answer the following questions about your car's symptoms:")
    print("1. Is the engine making strange noises? (yes/no)")
    answer = input().lower()
    if answer == "yes":
        symptoms.append("Strange engine noises")
        print("   - Is it a knocking sound or a squealing sound? (knocking/squealing)")
        engine_noise_type = input().lower()
        if engine_noise_type == "knocking":
            symptoms.append("Knocking engine noise")
        elif engine_noise_type == "squealing":
            symptoms.append("Squealing engine noise")
    print("2. Is the car experiencing rough idling? (yes/no)")
    answer = input().lower()
    if answer == "yes":
        symptoms.append("Rough idling")
        print("   - Is the idle RPM too high or too low? (high/low)")
        idling_issue = input().lower()
        if idling_issue == "high":
            symptoms.append("High idle RPM")
        elif idling_issue == "low":
            symptoms.append("Low idle RPM")

    print("3. Are you experiencing any issues with acceleration? (yes/no)")
    answer = input().lower()
    if answer == "yes":
        symptoms.append("Acceleration issues")
        print("   - Is the acceleration sluggish or jerky? (sluggish/jerky)")
        acceleration_issue = input().lower()
        if acceleration_issue == "sluggish":
            symptoms.append("Sluggish acceleration")
        elif acceleration_issue == "jerky":
            symptoms.append("Jerky acceleration")
    print("4. Is the car pulling to one side while driving? (yes/no)")
    answer = input().lower()
    if answer == "yes":
        symptoms.append("Pulling to one side")
        print("   - Is it pulling to the left or to the right? (left/right)")
        pulling_side = input().lower()
        if pulling_side == "left":
            symptoms.append("Pulling to the left")
        elif pulling_side == "right":
            symptoms.append("Pulling to the right")

    print("5. Is there smoke coming from the exhaust? (yes/no)")
    answer = input().lower()
    if answer == "yes":
        symptoms.append("Smoke from exhaust")
        print("   - What color is the smoke? (white/black/blue)")
        smoke_color = input().lower()
        if smoke_color == "white":
            symptoms.append("White smoke from exhaust")
        elif smoke_color == "black":
            symptoms.append("Black smoke from exhaust")
        elif smoke_color == "blue":
            symptoms.append("Blue smoke from exhaust")
    print("6. Are there any dashboard warning lights illuminated? (yes/no)")
    answer = input().lower()
    if answer == "yes":
        symptoms.append("Dashboard warning lights")
        print("   - Which warning lights are illuminated? (ABS/Check Engine/Others)")
        warning_lights = input().lower()
        if "abs" in warning_lights:
            symptoms.append("ABS warning light")
        if "check engine" in warning_lights:
            symptoms.append("Check engine warning light")
    return symptoms

def diagnose_issues(symptoms):
    issues = []
    if "Strange engine noises" in symptoms:
        if "Knocking engine noise" in symptoms:
            issues.append("Possible engine knock, may need engine inspection.")
        if "Squealing engine noise" in symptoms:
            issues.append("Possible issues with belts or pulleys, check belt tension and condition.")
    if "Rough idling" in symptoms:
        if "High idle RPM" in symptoms:
            issues.append("Possible vacuum leak or throttle issue, check idle control valve.")
        if "Low idle RPM" in symptoms:
            issues.append("Possible issues with fuel delivery or spark plugs, check fuel pressure and ignition
system.")
    if "Acceleration issues" in symptoms:
        if "Sluggish acceleration" in symptoms:
            issues.append("Possible issues with fuel system, check fuel filter and injectors.")
        if "Jerky acceleration" in symptoms:
            issues.append("Possible transmission issues, check transmission fluid and components.")
    if "Pulling to one side" in symptoms:
        if "Pulling to the left" in symptoms or "Pulling to the right" in symptoms:
            issues.append("Possible alignment or tire issues, check tire pressure and alignment.")
    if "Smoke from exhaust" in symptoms:
        if "White smoke from exhaust" in symptoms:
            issues.append("Possible coolant leak, check coolant level and radiator for leaks.")
        if "Black smoke from exhaust" in symptoms:
            issues.append("Possible issues with fuel mixture, check air filter and fuel injectors.")
        if "Blue smoke from exhaust" in symptoms:
            issues.append("Possible oil burning, check for oil leaks and engine condition.")
    if "Dashboard warning lights" in symptoms:
        if "ABS warning light" in symptoms:
            issues.append("Possible issues with ABS system, check ABS sensors and module.")
        if "Check engine warning light" in symptoms:
            issues.append("Possible engine issues, perform diagnostic scan for fault codes.")
    if len(issues) == 0:
        issues.append("No specific issues detected based on provided symptoms. Further inspection may be needed.")
    return issues
symptoms = ask_symptoms()
issues = diagnose_issues(symptoms)
print("\nPossible car issues based on symptoms:")
for issue in issues:
    print("-", issue)
```

Output



```
Welcome to the Auto Maintenance Expert System!
Please answer the following questions about your car's symptoms:
1. Is the engine making strange noises? (yes/no)
yes
   - Is it a knocking sound or a squealing sound? (knocking/squealing)
knocking
2. Is the car experiencing rough idling? (yes/no)
no
3. Are you experiencing any issues with acceleration? (yes/no)
no
4. Is the car pulling to one side while driving? (yes/no)
yes
   - Is it pulling to the left or to the right? (left/right)
right
5. Is there smoke coming from the exhaust? (yes/no)
yes
   - What color is the smoke? (white/black/blue)
black
6. Are there any dashboard warning lights illuminated? (yes/no)
no

Possible car issues based on symptoms:
- Possible engine knock, may need engine inspection.
- Possible alignment or tire issues, check tire pressure and alignment.
- Possible issues with fuel mixture, check air filter and fuel injectors.
```

Conclusion: Implemented Expert System for 4-wheeler Automobile maintenance and successfully executed.