NED University of Engineering and Technology, Karachi

Medical Diagnosis System

Complex Engineering Analysis

AIES

Submitted to :
Sir Kashif
By:
Syeda Umme Kulsoom (SE - 22064)

1. Introduction

Reactive logic refers to the ability of a system to automatically respond to inputs by triggering appropriate actions. Such systems are commonly used in expert systems, health monitoring devices, automation, and decision-support applications. This lab focuses on implementing event-driven logic where the program evaluates input data and decides actions without human intervention. The activity demonstrates how conditional statements and rule-based thinking form the basis of intelligent behavior in software systems.

2. Theory

Event-driven decision logic is built on **IF–THEN** rules:
IF a certain condition is met in the input,
THEN the system generates a specific response.

This approach is widely used in medical diagnosis systems, sensor-based monitoring, robotics, and expert systems. It relies on conditional structures such as `if`, `elif`, `else`, and sometimes nested conditions for more accurate reasoning. By checking combinations of symptoms or sensor values, the system mimics human decision-making and provides relevant alerts or predictions. The core idea is that **data drives behavior**, creating autonomous and intelligent applications.

3. Procedure

In this lab, a **medical diagnosis system** was implemented using Python.
The steps followed were:

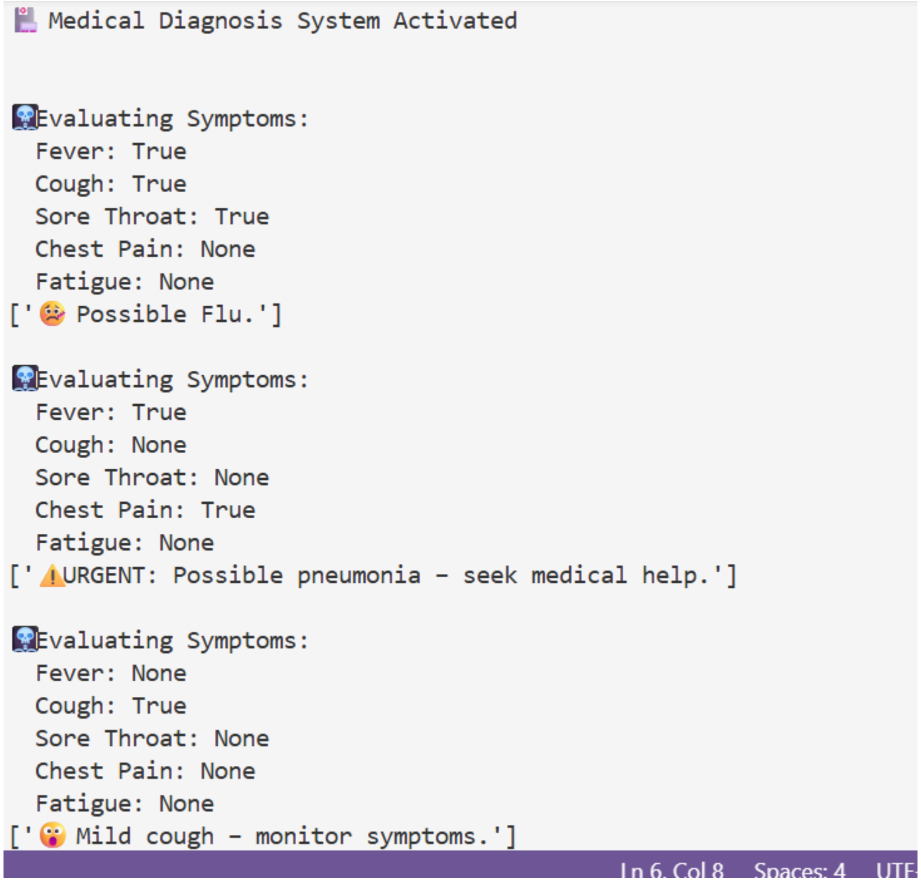
1. Defined a function `diagnose_patient()` that accepts symptoms as input.
2. Implemented nested conditional logic to evaluate combinations of symptoms.
3. Returned diagnosis actions instead of only printing them.
4. Executed multiple test scenarios to observe how the system responds to different symptom patterns.
5. Ensured the system can handle cases with no symptoms, mild symptoms, and high-risk indicators such as chest pain.

4. Code Snippet (Simplified)

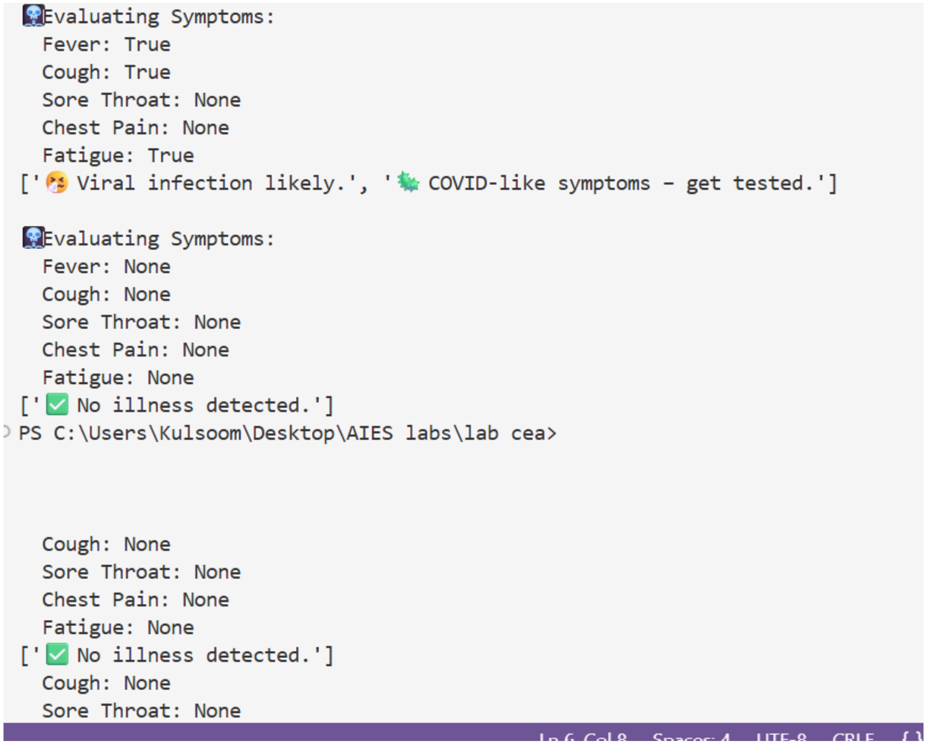
```python
def diagnose_patient(fever=None, cough=None, sore_throat=None,
                     chest_pain=None, fatigue=None):
    """
    Returns medical diagnosis based on symptoms.
    """
    actions = []

    print("\n Evaluating Symptoms:")
    print(f"  Fever: {fever}")
    print(f"  Cough: {cough}")
    print(f"  Sore Throat: {sore_throat}")
    print(f"  Chest Pain: {chest_pain}")
    print(f"  Fatigue: {fatigue}")

    if chest_pain:
        if fever or fatigue:
            actions.append("⚠️ URGENT: Possible pneumonia – seek medical help.")
        else:
            actions.append("⚠️ Warning: Chest pain detected – consult a doctor.")

    if fever:
        if cough and sore_throat:
            actions.append(" Possible Flu.")
        elif cough and fatigue:
            actions.append("Viral infection likely.")

    if fever and cough and fatigue:
        actions.append(" COVID-like symptoms – get tested.")

    if cough and not fever:
        if sore_throat:
            actions.append(" Possible throat infection.")
        else:
            actions.append(" Mild cough – monitor symptoms.")

    if not any([fever, cough, sore_throat, chest_pain, fatigue]):
        actions.append(" No illness detected.")

    return actions


if __name__ == "__main__":
```

```python
    print(" Medical Diagnosis System Activated\n")

    print(diagnose_patient(fever=True, cough=True, sore_throat=True))

    print(diagnose_patient(fever=True, chest_pain=True))

    print(diagnose_patient(cough=True))

    print(diagnose_patient(fever=True, cough=True, fatigue=True))

    print(diagnose_patient())
```

Output:



```
🏥 Medical Diagnosis System Activated


🩻Evaluating Symptoms:
  Fever: True
  Cough: True
  Sore Throat: True
  Chest Pain: None
  Fatigue: None
['🤒 Possible Flu.']

🩻Evaluating Symptoms:
  Fever: True
  Cough: None
  Sore Throat: None
  Chest Pain: True
  Fatigue: None
['⚠️URGENT: Possible pneumonia - seek medical help.']

🩻Evaluating Symptoms:
  Fever: None
  Cough: True
  Sore Throat: None
  Chest Pain: None
  Fatigue: None
['😮 Mild cough - monitor symptoms.']
```



```
🩻Evaluating Symptoms:
  Fever: True
  Cough: True
  Sore Throat: None
  Chest Pain: None
  Fatigue: True
['🤧 Viral infection likely.', '🦠 COVID-like symptoms - get tested.']

🩻Evaluating Symptoms:
  Fever: None
  Cough: None
  Sore Throat: None
  Chest Pain: None
  Fatigue: None
['✅ No illness detected.']
PS C:\Users\Kulsoom\Desktop\AIES labs\lab cea>



  Cough: None
  Sore Throat: None
  Chest Pain: None
  Fatigue: None
['✅ No illness detected.']
  Cough: None
  Sore Throat: None
```

5. Results

Scenario outputs included:

- Fever + cough + throat pain → *"Possible Flu"*
- Chest pain + fever → *"Possible pneumonia – urgent"*
- Only cough → *"Mild cough or throat infection"*
- Fever + cough + fatigue → *"COVID-like symptoms"*
- No symptoms → *"No illness detected"*

The system successfully produced different decisions based on input combinations, demonstrating reactive intelligence.

6. Conclusion

This lab demonstrated how reactive logic can be used to create an intelligent medical diagnosis system. By applying nested conditions and rule-based decision making, the program autonomously evaluates symptoms and returns meaningful diagnoses. The exercise showed how simple programming constructs can model real-world expert systems and support complex decision processes in healthcare and automation.