NAME: LAKSHANIKA VS
REG. NO:241801131
EXP.NO:10
EXP.NAME: FUZZY LOGIC

Commands   + Code   + Text

```
[1] !pip install scikit-fuzzy
```

```
Collecting scikit-fuzzy
  Downloading scikit_fuzzy-0.5.0-py2.py3-none-any.whl.metadata (2.6 kB)
  Downloading scikit_fuzzy-0.5.0-py2.py3-none-any.whl (920 kB)
  ──────────────────────────── 920.8/920.8 kB 16.5 MB/s eta 0:00:00
Installing collected packages: scikit-fuzzy
Successfully installed scikit-fuzzy-0.5.0
```

```python
[2] import numpy as np
    import skfuzzy as fuzz
    from skfuzzy import control as ctrl

    # Define fuzzy variables
    experience = ctrl.Antecedent(np.arange(0, 21, 1), 'experience')
    success_rate = ctrl.Antecedent(np.arange(0, 101, 1), 'success_rate')
    performance = ctrl.Consequent(np.arange(0, 101, 1), 'performance')

    # Define fuzzy membership functions
    experience['low'] = fuzz.trimf(experience.universe, [0, 0, 10])
    experience['medium'] = fuzz.trimf(experience.universe, [5, 10, 15])
    experience['high'] = fuzz.trimf(experience.universe, [10, 20, 20])

    success_rate['low'] = fuzz.trimf(success_rate.universe, [0, 0, 50])
    success_rate['medium'] = fuzz.trimf(success_rate.universe, [25, 50, 75])
    success_rate['high'] = fuzz.trimf(success_rate.universe, [50, 100, 100])

    performance['poor'] = fuzz.trimf(performance.universe, [0, 0, 50])
    performance['average'] = fuzz.trimf(performance.universe, [25, 50, 75])
    performance['excellent'] = fuzz.trimf(performance.universe, [50, 100, 100])

    # Define fuzzy rules
    rule1 = ctrl.Rule(experience['low'] & success_rate['low'], performance['poor'])
    rule2 = ctrl.Rule(experience['medium'] | success_rate['medium'], performance['average'])
    rule3 = ctrl.Rule(experience['high'] & success_rate['high'], performance['excellent'])

    # Create FIS control system
    performance_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])
    performance_sim = ctrl.ControlSystemSimulation(performance_ctrl)
```

```python
    performance['excellent'] = fuzz.trimf(performance.universe, [50, 100, 100])

    # Define fuzzy rules
    rule1 = ctrl.Rule(experience['low'] & success_rate['low'], performance['poor'])
    rule2 = ctrl.Rule(experience['medium'] | success_rate['medium'], performance['average'])
    rule3 = ctrl.Rule(experience['high'] & success_rate['high'], performance['excellent'])

    # Create FIS control system
    performance_ctrl = ctrl.ControlSystem([rule1, rule2, rule3])
    performance_sim = ctrl.ControlSystemSimulation(performance_ctrl)

    # Provide input values
    performance_sim.input['experience'] = 12
    performance_sim.input['success_rate'] = 70

    # Compute fuzzy inference
    performance_sim.compute()

    # Print the output
    print(f"Predicted Performance Score: {performance_sim.output['performance']:.2f}")
```

```
Predicted Performance Score: 57.52
```

Start coding or generate with AI.