Name: **GANESH SANJAY PANDHRE**

Roll Nos : **41** | Div: **D20B**

AIM: To implement Fuzzy Membership Functions.

Experiment No: 6

Theory

Introduction to Fuzzy Logic & Membership Functions

Fuzzy logic is a mathematical approach that deals with reasoning that is approximate rather than fixed and exact.
Unlike traditional binary logic (**True/False**, **1/0**), fuzzy logic allows for **degrees of truth** represented by values between 0 and 1.

A **membership function (MF)** defines how each input value is mapped to a membership degree (between 0 and 1) in a fuzzy set.

Examples of fuzzy set usage:

- **Temperature classification** (Cold, Warm, Hot)
- **Traffic density** (Light, Moderate, Heavy)
- **Speed control** (Slow, Medium, Fast)

Types of Membership Functions

---

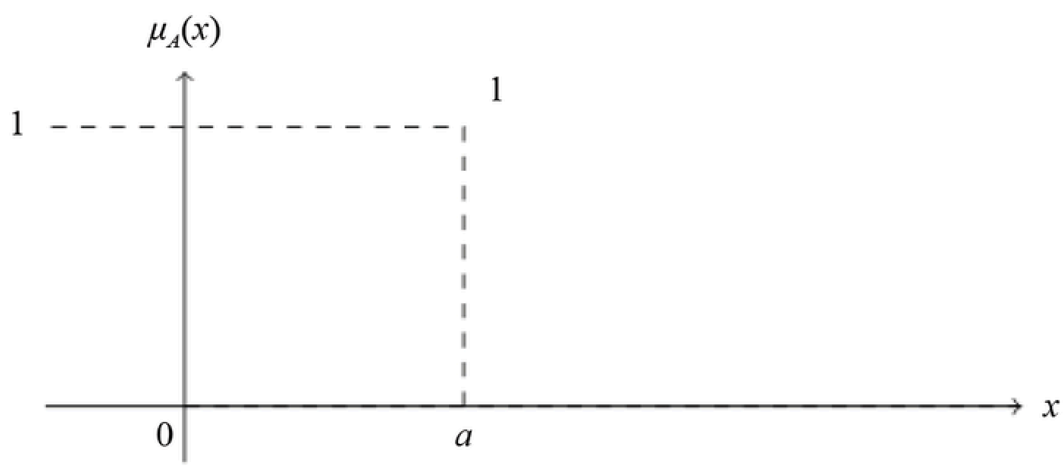
1. Singleton Membership Function

- Simplest form — only **one specific value** has full membership (1), and all others have zero membership.
- **Use Case:** Detecting an exact event (e.g., emergency vehicle presence in traffic).

Equation:

$$\mu(x) = \begin{cases} 1 & \text{if } x = x_0 \\ 0 & \text{otherwise} \end{cases}$$

Graph:



---

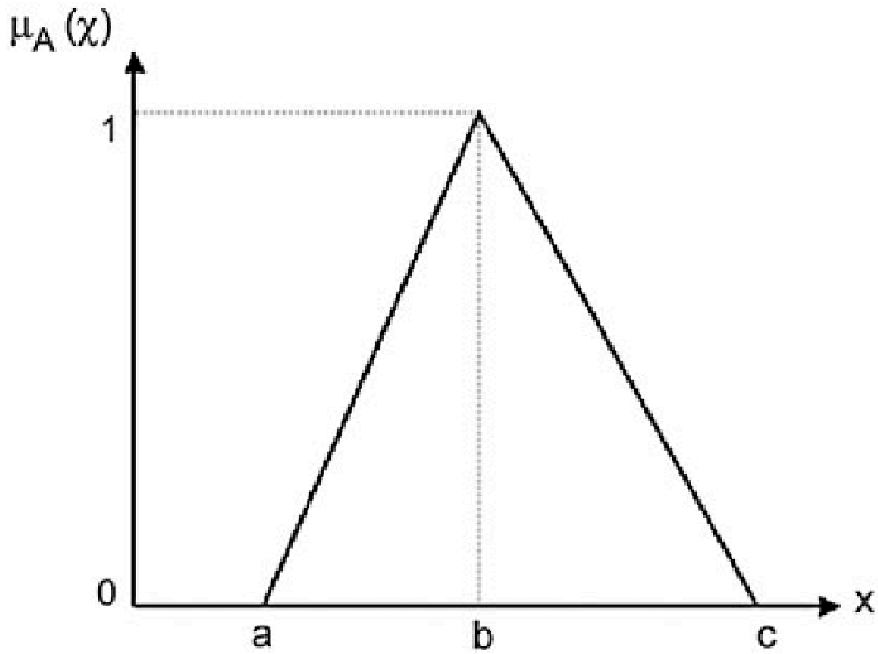
2. Triangular Membership Function

- Defined by three points **(a, b, c)** forming a triangle.
- **Use Case:** Representing moderate conditions (e.g., "medium speed").

Equation:

$$\mu(x) = \begin{cases} 0 & \text{if } x \leq a \text{ or } x \geq c \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ \frac{c-x}{c-b} & \text{if } b \leq x \leq c \end{cases}$$

Graph:



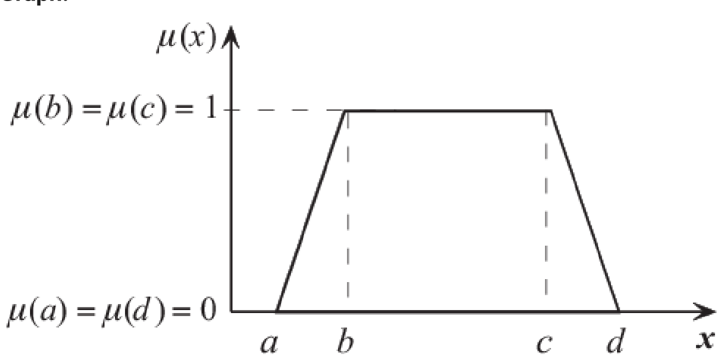
3. Trapezoidal Membership Function

- Similar to triangular but with a **flat top** (parameters: a, b, c, d).
- **Use Case:** Representing ranges with **full membership** (e.g., "heavy traffic" between 60–80 vehicles/min).

Equation:

$$\mu(x) = \begin{cases} 0 & \text{if } x \leq a \text{ or } x \geq d \\ \frac{x-a}{b-a} & \text{if } a \leq x \leq b \\ 1 & \text{if } b \leq x \leq c \\ \frac{d-x}{d-c} & \text{if } c \leq x \leq d \end{cases}$$

Graph:



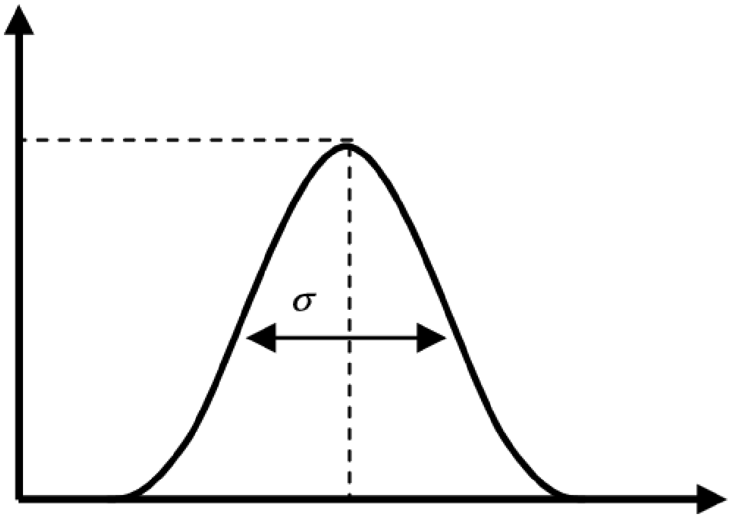
4. Gaussian Membership Function

- Smooth **bell-shaped curve** centered at mean (c) with spread (σ).
- **Use Case:** Natural variations (e.g., "light traffic" around 10 vehicles/min).

Equation:

$$\mu(x) = e^{-\frac{(x-c)^2}{2\sigma^2}}$$

Graph:



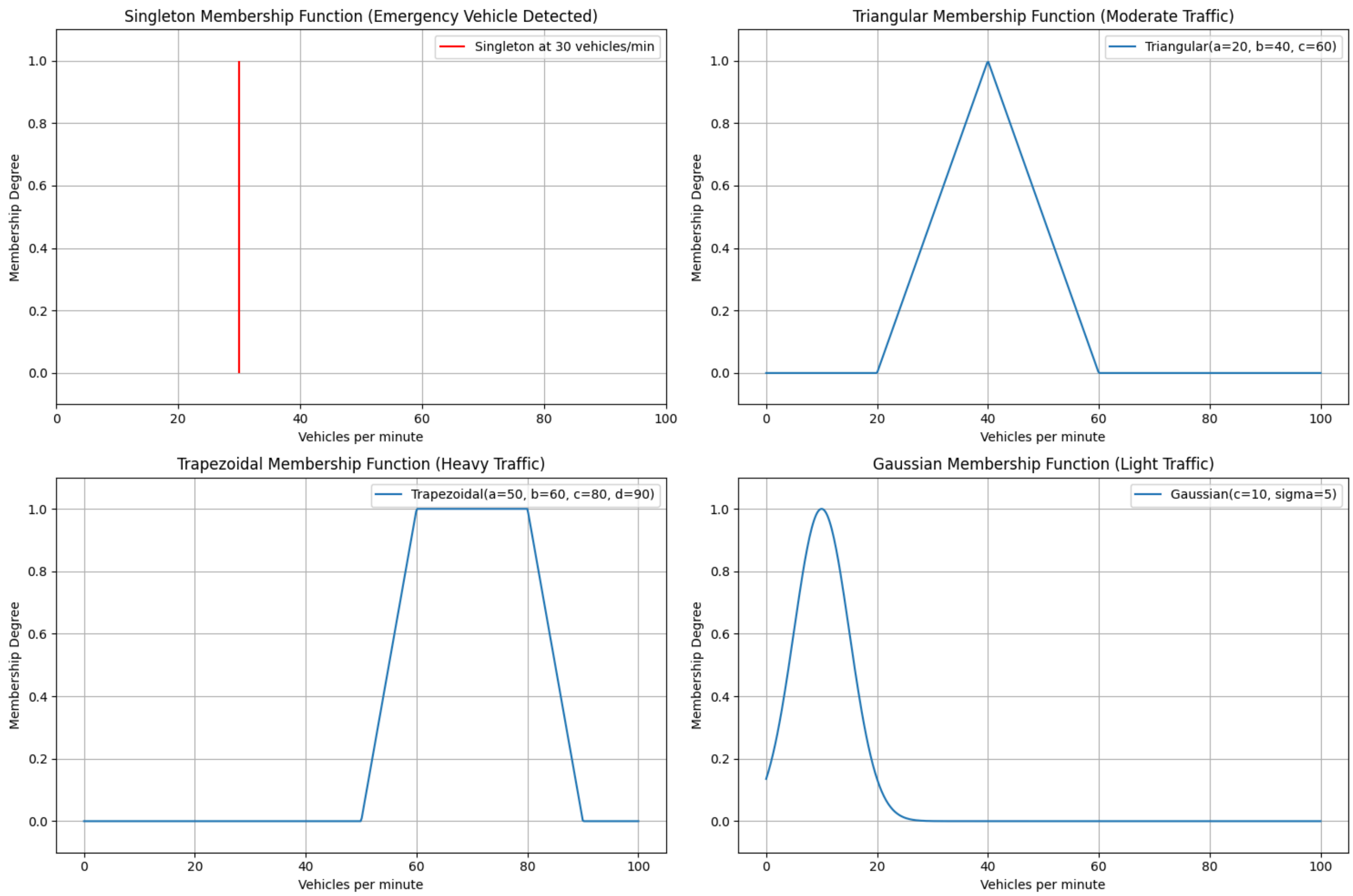
Applications of Membership Functions

- **Traffic Light Control:** Adjusting signal timings based on fuzzy traffic density.
- **Washing Machines:** Adjusting wash cycles based on dirt level (low, medium, high).
- **Air Conditioners:** Regulating temperature based on "comfort level" (cool, warm, hot).

```
# Import required libraries
import numpy as np
import matplotlib.pyplot as plt

# Define the membership functions
def singleton_mf(x, xo):
    return np.where(x == xo, 1, 0)

def triangular_mf(x, a, b, c):
    return np.maximum(np.minimum((x-a)/(b-a), (c-x)/(c-b)), 0)

def trapezoidal_mf(x, a, b, c, d):
    return np.maximum(np.minimum(np.minimum((x-a)/(b-a), 1), (d-x)/(d-c)), 0)

def gaussian_mf(x, c, sigma):
    return np.exp(-((x-c)**2) / (2*sigma**2))

# Create x values for traffic flow (vehicles per minute)
x_traffic = np.linspace(0, 100, 500)  # From 0 to 100 vehicles per minute
```

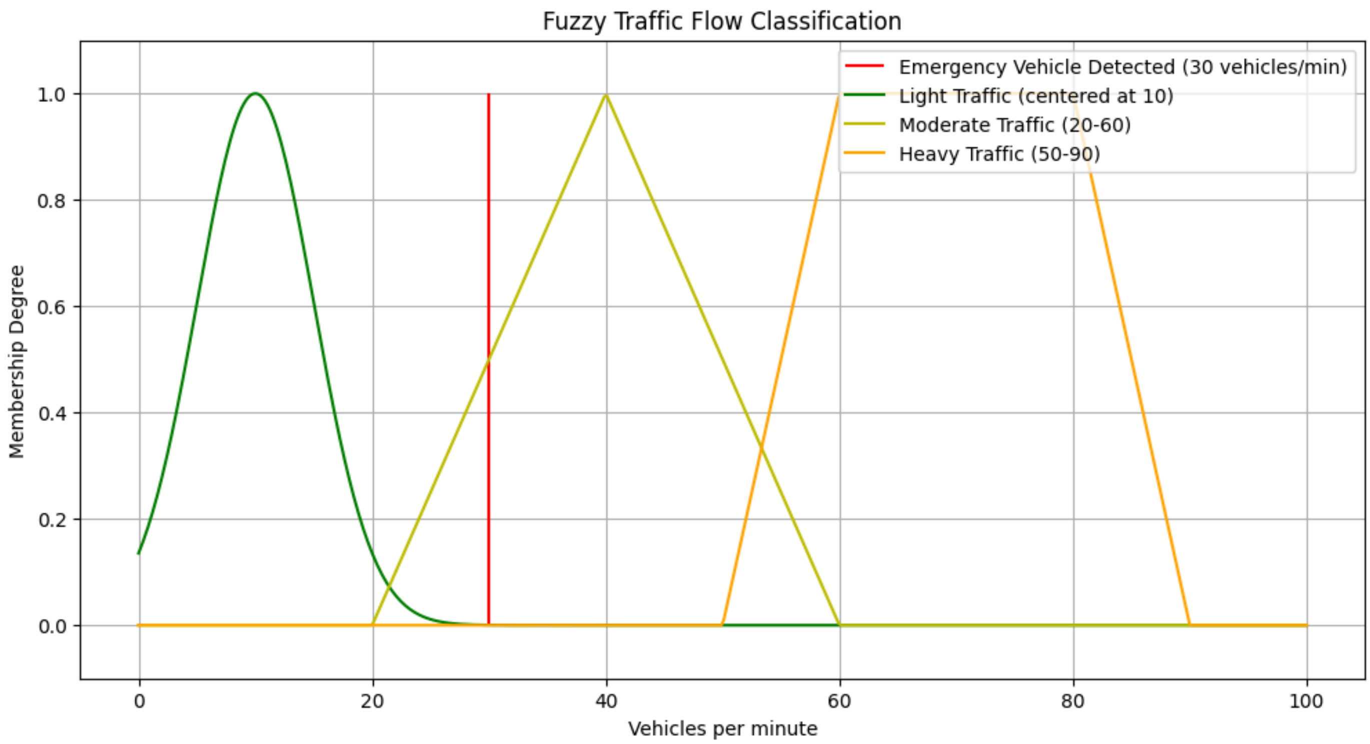
```
# Plot settings
plt.figure(figsize=(15, 10))

# Singleton Membership Function
plt.subplot(2, 2, 1)
plt.vlines(30, 0, 1, colors='r', label='Singleton at 30 vehicles/min')
plt.title("Singleton Membership Function (Emergency Vehicle Detected)")
plt.xlabel("Vehicles per minute")
plt.ylabel("Membership Degree")
plt.ylim(-0.1, 1.1)
plt.xlim(0, 100)
plt.legend()
plt.grid(True)

# Triangular Membership Function
plt.subplot(2, 2, 2)
plt.plot(x_traffic, triangular_mf(x_traffic, 20, 40, 60),
         label='Triangular(a=20, b=40, c=60)')
plt.title("Triangular Membership Function (Moderate Traffic)")
plt.xlabel("Vehicles per minute")
plt.ylabel("Membership Degree")
```

```
plt.ylabel("Membership Degree")
plt.ylim(-0.1, 1.1)
plt.legend()
plt.grid(True)

# Trapezoidal Membership Function
plt.subplot(2, 2, 3)
plt.plot(x_traffic, trapezoidal_mf(x_traffic, 50, 60, 80, 90),
         label='Trapezoidal(a=50, b=60, c=80, d=90)')
plt.title("Trapezoidal Membership Function (Heavy Traffic)")
plt.xlabel("Vehicles per minute")
plt.ylabel("Membership Degree")
plt.ylim(-0.1, 1.1)
plt.legend()
plt.grid(True)

# Gaussian Membership Function
plt.subplot(2, 2, 4)
plt.plot(x_traffic, gaussian_mf(x_traffic, 10, 5),
         label='Gaussian(c=10, sigma=5)')
plt.title("Gaussian Membership Function (Light Traffic)")
plt.xlabel("Vehicles per minute")
plt.ylabel("Membership Degree")
plt.ylim(-0.1, 1.1)
plt.legend()
plt.grid(True)

plt.tight_layout()
plt.show()
```




```
# Define traffic membership functions
emergency_vehicle = singleton_mf(x_traffic, 30)  # Singleton for emergency vehicle detection
light_traffic = gaussian_mf(x_traffic, 10, 5)    # Light traffic centered at 10 vehicles/min
moderate_traffic = triangular_mf(x_traffic, 20, 40, 60)  # Moderate traffic
heavy_traffic = trapezoidal_mf(x_traffic, 50, 60, 80, 90)  # Heavy traffic

# Plot combined traffic membership functions
plt.figure(figsize=(12, 6))

# Plot each membership function
plt.vlines(30, 0, 1, colors='red', linestyles='solid',
           label='Emergency Vehicle Detected (30 vehicles/min)')
plt.plot(x_traffic, light_traffic, 'g', label='Light Traffic (centered at 10)')
plt.plot(x_traffic, moderate_traffic, 'y', label='Moderate Traffic (20-60)')
plt.plot(x_traffic, heavy_traffic, 'orange', label='Heavy Traffic (50-90)')

# Formatting
plt.title('Fuzzy Traffic Flow Classification')
```

```
plt.xlabel('Vehicles per minute')
plt.ylabel('Membership Degree')
plt.ylim(-0.1, 1.1)
plt.legend(loc='upper right')
plt.grid(True)
plt.show()
```



Conclusion

Fuzzy membership functions help handle uncertainty by assigning degrees of truth (0 to 1) instead of strict true/false values. They are useful in real-world systems like traffic control, washing machines, and air conditioners, where decisions need flexibility. By implementing Singleton, Triangular, Trapezoidal, and Gaussian functions, we can model complex situations more naturally.

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```

```
Start coding or generate with AI.
```