Name: Swapnil De Sarkar Student ID: 231003004031

Assignment-4

```
# Implementation of Trapezoidal Fuzzy Sets
import matplotlib.pyplot as plt
import numpy as np

# 1. Class to represent a point
class Point:
  def __init__(self, x, y):
    self.x = x
    self.y = y

  def __str__(self):
    return f"({self.x}, {self.y})"

# 2. Class to represent a 2D straight line
class Line:
  def __init__(self, p1, p2):
    self.p1 = p1
    self.p2 = p2

  def slope(self):
    if self.p2.x - self.p1.x == 0:
      return float('inf')
    return (self.p2.y - self.p1.y) / (self.p2.x - self.p1.x)

  def y_for_x(self, x):
    m = self.slope()
    if m == float('inf'):
      return None
    return self.p1.y + m * (x - self.p1.x)

# 3. Function to display a straight line
def display_line(line, label=""):
  x = np.linspace(min(line.p1.x, line.p2.x), max(line.p1.x, line.p2.x), 100)
  y = [line.y_for_x(xi) for xi in x]
  plt.plot(x, y, label=label)
```

```
# 4. Class to represent a fuzzy set
class Fuzzy_set:
  def __init__(self, left_bottom, left_top, right_top, right_bottom):
    self.left_bottom = Point(*left_bottom)
    self.left_top = Point(*left_top)
    self.right_top = Point(*right_top)
    self.right_bottom = Point(*right_bottom)

  # 5. Function to display the fuzzy set
  def display(self, label=""):
    x = [self.left_bottom.x, self.left_top.x, self.right_top.x, self.right_bottom.x]
    y = [self.left_bottom.y, self.left_top.y, self.right_top.y, self.right_bottom.y]
    plt.plot(x, y, label=label)
    plt.fill(x, y, alpha=0.3)

  # 6. Function to return the complement of the fuzzy set
  def complement(self):
    return Fuzzy_set(
      (self.left_bottom.x, 1 - self.left_bottom.y),
      (self.left_top.x, 1 - self.left_top.y),
      (self.right_top.x, 1 - self.right_top.y),
      (self.right_bottom.x, 1 - self.right_bottom.y)
    )

  # 7. Function to calculate the membership of a point in the fuzzy set
  def membership(self, x):
    if x < self.left_bottom.x or x > self.right_bottom.x:
      return 0

    if self.left_top.x <= x <= self.right_top.x:
      return 1

    if x < self.left_top.x:
      left_line = Line(self.left_bottom, self.left_top)
      return left_line.y_for_x(x)

    right_line = Line(self.right_top, self.right_bottom)
    return right_line.y_for_x(x)
```

```
def main():
    # 3. Display two lines
    line1 = Line(Point(0, 4), Point(4, 0))
    line2 = Line(Point(0, 0), Point(4, 4))

    plt.figure(figsize=(10, 5))
    plt.subplot(1, 2, 1)
    display_line(line1, "Line 1")
    display_line(line2, "Line 2")
    plt.grid(True)
    plt.legend()
    plt.title("Two Lines")

    # 6. Create and display fuzzy set
    fs = Fuzzy_set((20, 0), (40, 1), (50, 1), (70, 0))
    fsc = fs.complement()

    plt.subplot(1, 2, 2)
    fs.display("Original Set")
    fsc.display("Complement Set")
    plt.grid(True)
    plt.legend()
    plt.title("Fuzzy Set and its Complement")

    plt.tight_layout()
    plt.show()

    # 8. Calculate memberships
    test_points = [20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70]
    print("\nMembership values:")
    print("Point\tOriginal\tComplement")
    print("-" * 40)
    for x in test_points:
        orig = fs.membership(x)
        comp = fsc.membership(x)
        print(f"{x}\t{orig:.2f}\t\t{comp:.2f}")
```

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

Output:

Membership values:

Point	Original	Complement
20	0.00	1.00
25	0.25	0.75
30	0.50	0.50
35	0.75	0.25
40	1.00	1.00
45	1.00	1.00
50	1.00	1.00
55	0.75	0.25
60	0.50	0.50
65	0.25	0.75
70	0.00	1.00

Name: Swapnil De Sarkar Student ID: 231003004031