**Tribhuvan University**

**Institute of Science and Technology**

****

**Central Department of Computer Science and Information Technology**

**Kirtipur, Kathmandu**

**Computational Geometry**

**Assignment**

**Lab 1: Implementation of Geometric Objects**

**Submitted by: Submitted To:**

**Name:** Raju Shrestha Asst. Prof. Jagdish Bhatta

**Roll No.:** 48/2079 CDCSIT

**Date:** 01-October-2023

**Lab 1:** Write a program to implement the following geometric objects.

1. Point
2. Line Segment
3. Ray
4. Line
5. Program to Implement Point

import matplotlib.pyplot as plt

class Node:

    def \_\_init\_\_(self, x, y):

        self.x = x

        self.y = y

        self.next = None

class LinkedList:

    def \_\_init\_\_(self):

        self.head = None

    def add\_point(self, x, y):

        new\_point = Node(x, y)

        new\_point.next = self.head

        self.head = new\_point

    def display\_points(self):

        current = self.head

        while current:

            print(f"({current.x}, {current.y})")

            current = current.next

    def get\_points(self):

        points = []

        current = self.head

        while current:

            points.append((current.x, current.y))

            current = current.next

        return points

list = LinkedList()

num\_points = int(input("Number of points: "))

for \_ in range(num\_points):

    x = float(input("Enter x coordinate: "))

    y = float(input("Enter y coordinate: "))

    list.add\_point(x, y)

list.display\_points()

points = list.get\_points()

x, y = zip(\*points)

plt.scatter(x, y, color='blue')

plt.title('Plot Data Points')

plt.xlabel('X-Coordinates')

plt.ylabel('Y-Coordinates')

plt.grid(True)

**Output:**

Number of points: 2

Enter x coordinate: 2

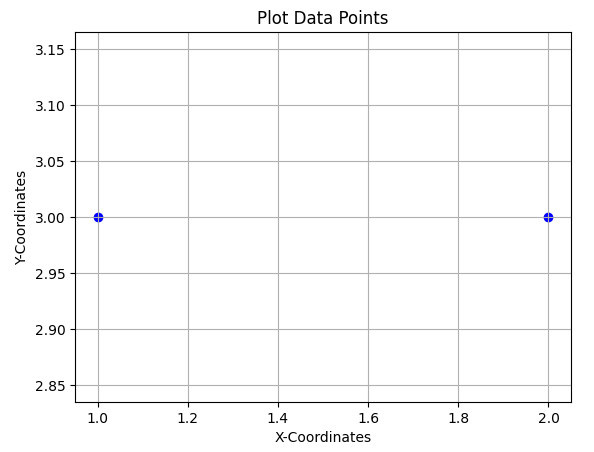
Enter y coordinate: 3

Enter x coordinate: 1

Enter y coordinate: 3

(1.0, 3.0)

(2.0, 3.0)



1. Program to Implement Line Segment

import matplotlib.pyplot as plt

class Node:

    def \_\_init\_\_(self, x, y):

        self.x = x

        self.y = y

        self.next = None

class LinkedList:

    def \_\_init\_\_(self):

        self.head = None

    def add\_point(self, x, y):

        new\_point = Node(x, y)

        new\_point.next = self.head

        self.head = new\_point

    def display\_points(self):

        current = self.head

        while current:

            print(f"({current.x}, {current.y})")

            current = current.next

    def get\_points(self):

        points = []

        current = self.head

        while current:

            points.append((current.x, current.y))

            current = current.next

        return points

list = LinkedList()

for \_ in range(int(input("Number of points: "))):

    list.add\_point(float(input("Enter x-coordinate: ")), float(input("Enter y-coordinate: ")))

list.display\_points()

points = list.get\_points()

x, y = zip(\*points)

plt.scatter(x, y, color='blue')

plt.plot(x, y, linestyle='-', color='red', marker='o')

plt.title('Plot Data Points with Line Segments')

plt.xlabel('X-Coordinates')

plt.ylabel('Y-Coordinates')

plt.grid(True)

plt.show()

**Output:**

Number of points: 2

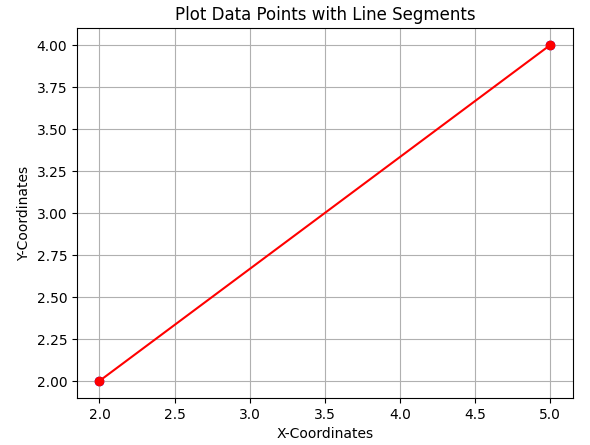
Enter x-coordinate: 2

Enter y-coordinate: 2

Enter x-coordinate: 5

Enter y-coordinate: 4

(5.0, 4.0)

(2.0, 2.0) 

1. Program to Implement Ray

import matplotlib.pyplot as plt

class Node:

    def \_\_init\_\_(self, x, y):

        self.x = x

        self.y = y

        self.next = None

class LinkedList:

    def \_\_init\_\_(self):

        self.head = None

    def add\_point(self, x, y):

        new\_point = Node(x, y)

        new\_point.next = self.head

        self.head = new\_point

    def display\_points(self):

        current = self.head

        while current:

            print(f"({current.x}, {current.y})")

            current = current.next

    def get\_points(self):

        points = []

        current = self.head

        while current:

            points.append((current.x, current.y))

            current = current.next

        return points

list = LinkedList()

for \_ in range(int(input("Number of points: "))):

    list.add\_point(float(input("Enter x-coordinate: ")), float(input("Enter y-coordinate: ")))

list.display\_points()

points = list.get\_points()

x, y = zip(\*points)

plt.scatter(x, y, color='blue')

common\_origin\_x, common\_origin\_y = x[0], y[0]

for i in range(1, len(x)):

    plt.quiver(common\_origin\_x, common\_origin\_y, x[i] - common\_origin\_x, y[i] - common\_origin\_y,

               angles='xy', scale\_units='xy', scale=1, color='red', width=0.005)

plt.title('Plot Data Points with Rays')

plt.xlabel('X-Coordinates')

plt.ylabel('Y-Coordinates')

plt.grid(True)

plt.show()

**Output:**

Number of points: 2

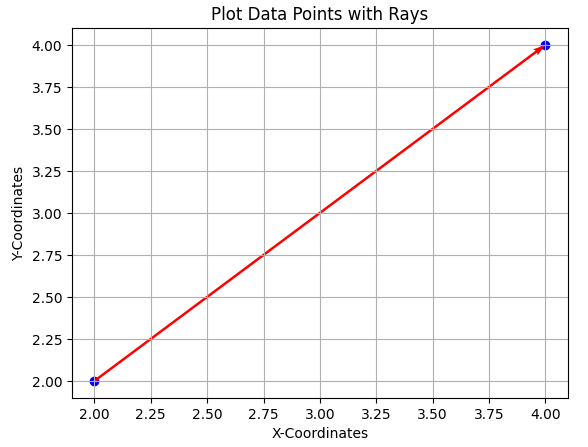
Enter x-coordinate: 4

Enter y-coordinate: 4

Enter x-coordinate: 2

Enter y-coordinate: 2

(2.0, 2.0)

(4.0, 4.0) 

1. Program to Implement Line

import matplotlib.pyplot as plt

class Node:

    def \_\_init\_\_(self, x, y):

        self.x = x

        self.y = y

        self.next = None

class LinkedList:

    def \_\_init\_\_(self):

        self.head = None

    def add\_point(self, x, y):

        new\_point = Node(x, y)

        new\_point.next = self.head

        self.head = new\_point

    def display\_points(self):

        current = self.head

        while current:

            print(f"({current.x}, {current.y})")

            current = current.next

    def get\_points(self):

        points = []

        current = self.head

        while current:

            points.append((current.x, current.y))

            current = current.next

        return points

list = LinkedList()

num\_points = int(input("Number of points: "))

for i in range(1, num\_points + 1):

    x\_coord = float(input(f"Enter x-coordinate for P{i}: "))

    y\_coord = float(input(f"Enter y-coordinate for P{i}: "))

    list.add\_point(x\_coord, y\_coord)

list.display\_points()

points = list.get\_points()

x, y = zip(\*points)

center\_x, center\_y = sum(x) / len(x), sum(y) / len(y)

plt.scatter(x, y, color='blue')

for i in range(len(x)):

    end\_x = center\_x + (center\_x - x[i]) \* 5

    end\_y = center\_y + (center\_y - y[i]) \* 5

    plt.plot([center\_x, end\_x], [center\_y, end\_y], linestyle='-', color='green')

    plt.annotate(f'P{i}', (x[i], y[i]), textcoords="offset points", xytext=(0, 5), ha='center', fontsize=8)

plt.title('Plot Data Points with Lines')

plt.xlabel('X-Coordinates')

plt.ylabel('Y-Coordinates')

plt.grid(True)

plt.show()

**Output:**

Number of points: 2

Enter x-coordinate for P1: 2

Enter y-coordinate for P1: 2

Enter x-coordinate for P2: 6

Enter y-coordinate for P2: 6

(6.0, 6.0)

(2.0, 2.0) 