**Question 1:**

**Code:**

import matplotlib.pyplot as plt

import numpy as np

1#st method

def plot\_multiple\_lines():

  N = int(input("Enter the number of Flights: "))

  Flights = []

  for i in range(N):

    line\_str = input(f"Enter coordinates for Flight {i+1} (e.g., 1,1 2,4 3,2): ")

    coords = [(int(x), int(y)) for x, y in [pair.split(',') for pair in line\_str.split()]]

    Flights.append(coords)

  plt.figure(figsize=(8, 6))

  for i, line in enumerate(Flights):

    xs = [x[0] for x in line]

    ys = [x[1] for x in line]

    plt.plot(xs, ys, linestyle='solid', linewidth=3, marker='o', markerfacecolor='blue', markersize=12)

  plt.xlabel("X-axis")

  plt.ylabel("Y-axis")

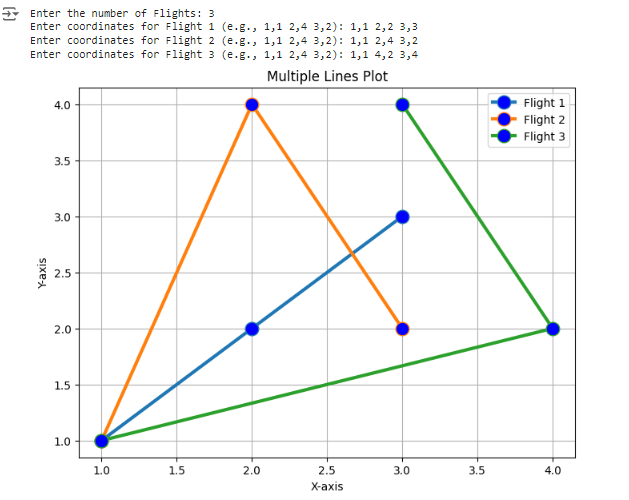
  plt.title("Multiple Lines Plot")

  plt.grid(True)

  plt.legend([f'Flights {i+1}' for i in range(N)])

  plt.show()

plot\_multiple\_lines()



References

Prompt: plot the graph by using list type corrdinate like (1,1),(2,2),(3,3)

<https://stackoverflow.com/questions/21519203/plotting-a-list-of-x-y-coordinates>

prompt: ploting graph with line and make cordinates dot in python

<https://www.geeksforgeeks.org/graph-plotting-in-python-set-1/>

<https://www.geeksforgeeks.org/linestyles-in-matplotlib-python/>

prompt: Python intersection of two splines

<https://stackoverflow.com/questions/42464334/find-the-intersection-of-two-curves-given-by-x-y-data-with-high-precision-in>

Code:

#2nd method

import matplotlib.pyplot as plt

import numpy as np

def plot\_lines(list\_of\_lines):

  plt.figure(figsize=(8, 6))

  for i, li in enumerate(list\_of\_lines):

    xs = [x[0] for x in li]

    ys = [x[1] for x in li]

    plt.plot(xs, ys, linestyle='solid', linewidth=3,

             marker='o', markersize=12, label=f'Line {i+1}')

  plt.xlabel("X-axis")

  plt.ylabel("Y-axis")

  plt.title("Flight routes ")

  plt.grid(True)

  plt.legend()

  plt.show()

# Get user input for the number of lines

N = int(input("Enter the number of lines: "))

# Initialize an empty list to store the lines

list\_of\_lines = []

# Get coordinates for each line from the user

for i in range(N):

  line = []

  num\_points = int(input(f"Enter the number of points for line {i+1}: "))

  for j in range(num\_points):

    x = int(input(f"Enter x-coordinate for point {j+1}: "))

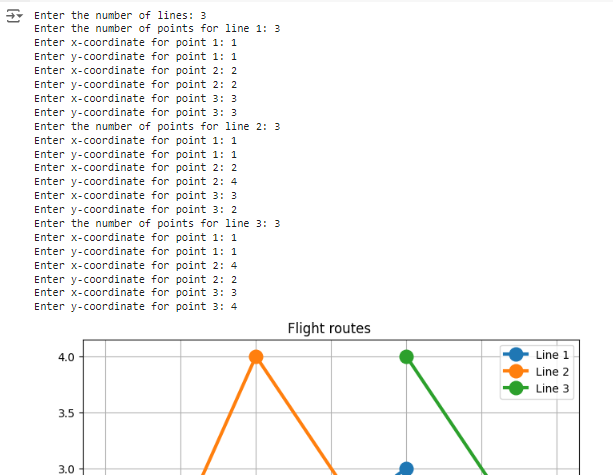
    y = int(input(f"Enter y-coordinate for point {j+1}: "))

    line.append((x, y))

  list\_of\_lines.append(line)

# Plot the lines

plot\_lines(list\_of\_lines)



A graph with lines and dots

Description automatically generated

**Question 2:**

**Code:**

# Weights in grams

apple\_weights = []

while True:

  user\_input = int(input("Enter apple weight in gram (-1 to stop ) : "))

  if user\_input == -1:

    break

  apple\_weights.append(user\_input)

total\_cost = 100  # Rupees

ram\_paid = 50

shyam\_paid = 30

rahim\_paid = 20

#total\_weight=sum(apple\_weights)

# Calculate the proportion of apples each person should get

ram\_proportion = (ram\_paid / total\_cost) \* total\_weight

shyam\_proportion = (shyam\_paid / total\_cost) \* total\_weight

rahim\_proportion = (rahim\_paid / total\_cost) \* total\_weight

#print(ram\_proportion,shyam\_proportion,rahim\_proportion)

# Sort the apples by weight (descending)

apple\_weights.sort(reverse=True)

#print(apple\_weights)

# Initialize lists to store the apples each person receives

ram\_apples = []

shyam\_apples = []

rahim\_apples = []

# Distribute apples

remaining\_apples = apple\_weights.copy()  # Create a copy to avoid modifying the original list

def distribute\_apples(person\_apples, proportion, remaining\_apples):

  i = 0

  while i < len(remaining\_apples):

    if remaining\_apples[i] <= proportion:

      person\_apples.append(remaining\_apples[i])

      proportion -= remaining\_apples[i]

      remaining\_apples.pop(i)  # Remove the apple from the remaining list

    else:

      i += 1  # Move to the next apple if it's too heavy

distribute\_apples(ram\_apples, ram\_proportion, remaining\_apples)

distribute\_apples(shyam\_apples, shyam\_proportion, remaining\_apples)

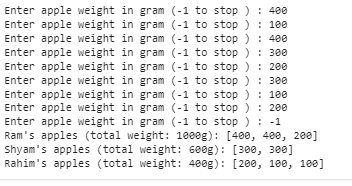
distribute\_apples(rahim\_apples, rahim\_proportion, remaining\_apples)

# Print results

print("Ram's apples (total weight: {}g): {}".format(sum(ram\_apples), ram\_apples))

print("Shyam's apples (total weight: {}g): {}".format(sum(shyam\_apples), shyam\_apples))

print("Rahim's apples (total weight: {}g): {}".format(sum(rahim\_apples), rahim\_apples))



References

Prompt:sorting the list in python

<https://www.w3schools.com/python/ref_list_sort.asp>

Prompt: making copy of list in python

<https://www.w3schools.com/python/python_lists_copy.asp>

**Question 3:**

**Code:**

**One more path is =**(1,2)kill->(1,1)kill->(4,1)kill->(4,2)kill->(8,2)kill->(8,9)kill->(5,9)Jump->(1,9)kill->(1,2)