In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
airport = pd.read_csv('D:/workurious/Airports-Only.csv', encoding='latin-1')
routes = pd.read_csv('D:Workurious/routes.csv')
planes = pd.read_csv('D:/Workurious/planes.csv', encoding='latin-1')
airlines = pd.read_csv('D:/Workurious/airlines.csv', encoding='latin-1')
t1 = pd.read_csv('D:/Workurious/table-1.csv', encoding='latin-1')
t2 = pd.read_csv('D:/Workurious/table-2.csv', encoding='latin-1')
t3 = pd.read_csv('D:/Workurious/table-3.csv', encoding='latin-1')
t4 = pd.read_csv('D:/Workurious/table-4.csv', encoding='latin-1')
t5 = pd.read_csv('D:/Workurious/table-5.csv', encoding='latin-1')
t6 = pd.read_csv('D:/Workurious/table-6.csv', encoding='latin-1')
```

In [3]:

```
input1 = input("Enter the first airport name : ")
print(input1)
```

```
Enter the first airport name : imphal
imphal
```

In [4]:

```
match1 = airport.loc[airport['City'].str.contains(input1, case=False, na = False)
match1
```

Out[4]:

	Airport_ID	Name	City	Country	IATA	ICAO	Latitude	Longitude	Altitude	Timezone	DS
2888	3050	Imphal Airport	Imphal	India	IMF	VEIM	24.76	93.896698	2540	5.5	

In [5]:

```
C1 = input("Enter the ICAO code of the source Airport from the options : ")
ID1 = input("Enter the Airport_ID of the source airport : ")
```

```
Enter the ICAO code of the source Airport from the options : VEIM
Enter the Airport_ID of the source airport : 3050
```

In [6]:

```
input2 = input("Enter the second airport name : ")
print(input2)
```

```
Enter the second airport name : mUMBAI
mUMBAI
```

In [7]:

```
match2 = airport.loc[airport['City'].str.contains(input2, case=False, na = False)
match2
```

Out[7]:

	Airport_ID	Name	City	Country	IATA	ICAO	Latitude	Longitude	Altitude	Timez
2839	2997	Chhatrapati Shivaji International Airport	Mumbai	India	BOM	VABB	19.088699	72.867897	39	

In [8]:

```
C2 = input("Enter the ICAO code of the destination Airport from the options : ")
ID2 = input("Enter the Airport_ID of the destination airport : ")
```

```
Enter the ICAO code of the destination Airport from the options : VABB
Enter the Airport_ID of the destination airport : 2997
```

In [9]:

```
arr1 = airport.to_numpy()
arr2 = routes.to_numpy()
arr3 = planes.to_numpy()
arr4 = t1.to_numpy()
arr5 = t2.to_numpy()
arr6 = t3.to_numpy()
arr7 = t4.to_numpy()
arr8 = t5.to_numpy()
```

In [10]:

```
r1 = np.where(arr1 == C1)
r2 = np.where(arr1 == C2)
```

In [11]:

```
La1 = arr1[r1[0],r1[1]+1]
Lo1 = arr1[r1[0],r1[1]+2]
La2 = arr1[r2[0],r2[1]+1]
Lo2 = arr1[r2[0],r2[1]+2]
```

In [12]:

```
import math

# Python 3 program for the
# haversine formula
def haversine(lat1, lon1, lat2, lon2):

    # distance between latitudes
    # and longitudes
    dLat = (lat2 - lat1) * math.pi / 180.0
    dLon = (lon2 - lon1) * math.pi / 180.0

    # convert to radians
    lat1 = (lat1) * math.pi / 180.0
    lat2 = (lat2) * math.pi / 180.0

    # apply formulae
    a = (pow(math.sin(dLat / 2), 2) +
         pow(math.sin(dLon / 2), 2) *
             math.cos(lat1) * math.cos(lat2));
    rad = 6371
    c = 2 * math.asin(math.sqrt(a))
    return rad * c

distance = haversine(La1, Lo1, La2, Lo2)
print(distance, "k.m.")
```

```
2256.074406550681 k.m.
```

In [13]:

```
r3 = np.where(arr2 == ID1)
r4 = np.where(arr2 == ID2)
```

In [14]:

```
m = len(r3[0])
print(m)
n = len(r4[0])
print(n)
```

```
18
452
```

In [15]:

```
flag = 0
for i in range(m):
    for j in range(n):
        if r3[0][i] == r4[0][j]:
            p = arr2[r3[0][i]][7]
            flag = 1
            break
if flag == 0:
    print("No route exist")
else:
    print("Route exist")
```

```
No route exist
```

In [16]:
```
print(p)
match3 = planes.loc[planes['IATA'].str.contains(p, case=False, na = False)]
match3
```

```
319
```

Out[16]:

	Name	IATA	ICAO
12	Airbus A319	319	A319

In [17]:
```
r3 = np.where(arr3 == p)
pname = arr3[r3[0], r3[1]-1]
pname
```

Out[17]:
```
array(['Airbus A319'], dtype=object)
```

In [18]:
```
r4 = np.where(arr4 == pname[0])
r5 = np.where(arr5 == pname[0])
r6 = np.where(arr6 == pname[0])
r7 = np.where(arr7 == pname[0])
r8 = np.where(arr8 == pname[0])
```

In [19]:
```
if distance < 926:
    fb = arr4[r4[0], r4[1]+3] * distance
    sc = arr4[r4[0], r4[1]+2]
elif distance <= 1267 and distance >= 926:
    fb = arr5[r5[0], r5[1]+4] * distance
    sc = arr5[r5[0], r5[1]+2]
elif distance < 3240 and distance >= 1900:
    fb = arr6[r6[0], r6[1]+3] * distance
    sc = arr6[r6[0], r6[1]+2]
elif distance <= 6300 and distance >= 3240:
    fb = arr7[r7[0], r7[1]+4] * distance
    sc = arr7[r7[0], r7[1]+2]
else:
    fb = arr8[r8[0], r8[1]+4] * distance
    sc = arr8[r8[0], r8[1]+2]
```

In [20]:
```
fb
```

Out[20]:
```
array([1764.3177586077156], dtype=object)
```

In [87]:
```
co2_total = 3.16*fb
co2_individual = co2_total/sc
print(co2_individual)
```

```
[44.961646106454694]
```

In [ ]:
```

```