Implementation of K-Nearest Neighbours (KNN)

1. Importing necessary libraries

First, we need to import the libraries required for KNN implementation. We will use `pandas` for data manipulation, `numpy` for numerical operations, and `Counter` (from `collections`) for label counting and finding the majority.

```
import pandas as pd
import numpy as np
from collections import Counter
```

2. Defining the relevant functions

Next, we define the necessary functions for the KNN algorithm. The `euclidean_distance` function computes the distance between two samples. The `get_neighbors` function finds the K-nearest neighbors by calculating distances from the test sample to all samples in the training set. Finally, the `predict_label` function determines the most common label among the neighbors.

```
def euclidean_distance(sample1, sample2):
    return np.sqrt(np.sum((sample1 - sample2) ** 2))

def get_neighbors(train, test_sample, k):
    distances = []

    for index, row in train.iterrows():
        distance = euclidean_distance(row[:-1].values, test_sample)
        distances.append((index, distance))

    distances.sort(key=lambda x: x[1])
    neighbors = distances[:k]

    return neighbors

def predict_label(neighbors, train):
    labels = [train.iloc[i[0], -1] for i in neighbors]
    majority_label = Counter(labels).most_common(1)[0][0]
    return majority_label
```

3. First Dataset: Age, Loan Amount, and Defaulter Status

Here, we create a dataset of 10 samples with features such as age, loan amount, and defaulter status. The user is prompted to input an unlabeled sample consisting of age and loan amount. We then apply the KNN algorithm, calculate the distances, display the K-nearest neighbors, and predict the label based on the majority vote.

```
data1 = {
    'Age': [25, 30, 35, 40, 22, 29, 41, 32, 36, 28],
    'Loan Amount': [5000, 10000, 15000, 20000, 3000, 12000, 25000, 17000, 9000, 11000],
    'Defaulter': ['No', 'No', 'Yes', 'Yes', 'No', 'No', 'Yes', 'Yes', 'No', 'No']
}

df1 = pd.DataFrame(data1)
print("Dataset 1:")
print(df1)

unlabeled_sample1 = np.array([int(input("Enter age: ")), int(input("Enter loan amount: "))])

k = 5
neighbors1 = get_neighbors(df1, unlabeled_sample1, k)

distances_df1 = pd.DataFrame(neighbors1, columns=['Index', 'Distance'])
print("Distances to the 5 nearest samples:")
print(distances_df1)

print("\nK-nearest neighbours: (K=5)")
neighbors_indices1 = [i[0] for i in neighbors1]
knn_df1 = df1.iloc[neighbors_indices1]
print(knn_df1)

predicted_label1 = predict_label(neighbors1, df1)
print(f"\nPredicted label for the unlabeled sample: {predicted_label1}")
```

```
Dataset 1:
   Age  Loan Amount Defaulter
0   25         5000        No
1   30        10000        No
2   35        15000       Yes
```

```
3   40        20000       Yes
4   22         3000        No
5   29        12000        No
6   41        25000       Yes
7   32        17000       Yes
8   36         9000        No
9   28        11000        No
Enter age: 30
Enter loan amount: 22000
Distances to the 5 nearest samples:
   Index      Distance
0      3   2000.025000
1      6   3000.020167
2      7   5000.000400
3      2   7000.001786
4      5  10000.000050

K-nearest neighbors: (K=5)
   Age  Loan Amount Defaulter
3   40        20000       Yes
6   41        25000       Yes
7   32        17000       Yes
2   35        15000       Yes
5   29        12000        No

Predicted label for the unlabeled sample: Yes
```

4. Second Dataset: Height, Weight, and T-shirt Size

Then, we create a dataset containing height, weight, and T-shirt size for 10 individuals. After prompting the user for an unlabeled sample consisting of height and weight, we implement the KNN algorithm, display distances, show the K-nearest neighbors, and predict the T-shirt size based on the nearest samples.

```
data2 = {
    'Height': [150, 160, 170, 180, 165, 175, 155, 158, 172, 168],
    'Weight': [50, 60, 70, 80, 65, 75, 55, 58, 72, 68],
    'T-shirt Size': ['S', 'M', 'L', 'L', 'M', 'L', 'S', 'S', 'M', 'M']
}

df2 = pd.DataFrame(data2)
print("Dataset 2:")
print(df2)

unlabeled_sample2 = np.array([int(input("Enter height (cm): ")), int(input("Enter weight (kg): "))])

neighbors2 = get_neighbors(df2, unlabeled_sample2, k)

distances_df2 = pd.DataFrame(neighbors2, columns=['Index', 'Distance'])
print("Distances to the 5 nearest samples:")
print(distances_df2)

print("\nK-nearest neighbours: (K=5)")
neighbors_indices2 = [i[0] for i in neighbors2]
knn_df2 = df2.iloc[neighbors_indices2]
print(knn_df2)

predicted_label2 = predict_label(neighbors2, df2)
print(f"\nPredicted label for the unlabeled sample: {predicted_label2}")
```

```
Dataset 2:
   Height  Weight T-shirt Size
0     150      50            S
1     160      60            M
2     170      70            L
3     180      80            L
4     165      65            M
5     175      75            L
6     155      55            S
7     158      58            S
8     172      72            M
9     168      68            M
Enter height (cm): 172
Enter weight (kg): 68
Distances to the 5 nearest samples:
   Index  Distance
0      2  2.828427
1      8  4.000000
2      9  4.000000
3      4  7.615773
4      5  7.615773

K-nearest neighbours: (K=5)
   Height  Weight T-shirt Size
2     170      70            L
8     172      72            M
9     168      68            M
4     165      65            M
5     175      75            L
```

```
Predicted label for the unlabeled sample: M
```

5. Third Dataset: Weight, Size, and Fruit Type

Finally, we create a dataset of 10 samples representing weight, size, and fruit type. The user provides an unlabeled sample with weight and size. We execute the KNN algorithm, calculate distances, display the K-nearest neighbors, and predict the fruit type based on the nearest neighbors.

```
data3 = {
    'Weight': [200, 150, 180, 220, 170, 190, 210, 160, 175, 185],
    'Size': [10, 8, 9, 11, 7, 10, 12, 8, 9, 10],
    'Fruit': ['Apple', 'Banana', 'Orange', 'Apple', 'Banana', 'Orange', 'Apple', 'Banana', 'Orange', 'Apple']
}

df3 = pd.DataFrame(data3)
print("Dataset 3:")
print(df3)

unlabeled_sample3 = np.array([int(input("Enter weight (g): ")), int(input("Enter size (cm): "))])

neighbors3 = get_neighbors(df3, unlabeled_sample3, k)

distances_df3 = pd.DataFrame(neighbors3, columns=['Index', 'Distance'])
print("Distances to the 5 nearest samples:")
print(distances_df3)

print("\nK-nearest neighbors: (K=5)")
neighbors_indices3 = [i[0] for i in neighbors3]
knn_df3 = df3.iloc[neighbors_indices3]
print(knn_df3)

predicted_label3 = predict_label(neighbors3, df3)
print(f"\nPredicted label for the unlabeled sample: {predicted_label3}")
```

```
Dataset 3:
   Weight  Size   Fruit
0     200    10   Apple
1     150     8  Banana
2     180     9  Orange
3     220    11   Apple
4     170     7  Banana
5     190    10  Orange
6     210    12   Apple
7     160     8  Banana
8     175     9  Orange
9     185    10   Apple
Enter weight (g): 160
Enter size (cm): 7
Distances to the 5 nearest samples:
   Index   Distance
0      7   1.000000
1      4  10.000000
2      1  10.049876
3      8  15.132746
4      2  20.099751

K-nearest neighbors: (K=5)
   Weight  Size   Fruit
7     160     8  Banana
4     170     7  Banana
1     150     8  Banana
8     175     9  Orange
2     180     9  Orange

Predicted label for the unlabeled sample: Banana
```