Latihan 1

```
import math
from collections import Counter
data = [
    {"Temperatur (°C)": 10, "Kecepatan Angin (km/jam)": 0,
"Persepsi": "Dingin"},
    {"Temperatur (°C)": 25, "Kecepatan Angin (km/jam)": 0,
"Persepsi": "Panas"},
    {"Temperatur (°C)": 15, "Kecepatan Angin (km/jam)": 5,
"Persepsi": "Dingin"},
    {"Temperatur (°C)": 20, "Kecepatan Angin (km/jam)": 3,
"Persepsi": "Panas"},
    {"Temperatur (°C)": 18, "Kecepatan Angin (km/jam)": 7,
"Persepsi": "Dingin"},
    {"Temperatur (°C)": 20, "Kecepatan Angin (km/jam)": 10,
"Persepsi": "Dingin"},
    {"Temperatur (°C)": 22, "Kecepatan Angin (km/jam)": 5,
"Persepsi": "Panas"},
    {"Temperatur (°C)": 24, "Kecepatan Angin (km/jam)": 6,
"Persepsi": "Panas"},
]

import pandas as pd
df = pd.DataFrame(data)
print("=== DATA TRAINING ===")
print(df)
```

```
=== DATA TRAINING ===
   Temperatur (°C)  Kecepatan Angin (km/jam) Persepsi
0               10                         0   Dingin
1               25                         0    Panas
2               15                         5   Dingin
3               20                         3    Panas
4               18                         7   Dingin
5               20                        10   Dingin
6               22                         5    Panas
7               24                         6    Panas
```

```
# menghitung Jarak
def distance(p1, p2):
    return math.sqrt((p1["Temperatur (°C)"] - p2["Temperatur
(°C)"])**2 +
                     (p1["Kecepatan Angin (km/jam)"] - p2["Kecepatan
Angin (km/jam)"])**2)

# KNN
def knn_predict(temp, wind, k):
```

```
    test_point = {"Temperatur (°C)": temp, "Kecepatan Angin (km/jam)":
wind}

    distances = []
    for row in data:
        dist = distance(test_point, row)
        distances.append((dist, row["Persepsi"]))

    distances.sort(key=lambda x: x[0])
    k_neighbors = [label for _, label in distances[:k]]
    return Counter(k_neighbors).most_common(1)[0][0]

# Prediksi Dataset
print("Prediksi Persepsi Marry untuk (16°C, 3 km/jam)")
for k in range(1, 6):
    pred = knn_predict(16, 3, k)
    print(f"k = {k} → Persepsi: {pred}")
```

```
Prediksi Persepsi Marry untuk (16°C, 3 km/jam)
k = 1 → Persepsi: Dingin
k = 2 → Persepsi: Dingin
k = 3 → Persepsi: Dingin
k = 4 → Persepsi: Dingin
k = 5 → Persepsi: Dingin
```

```
# Evaluasi Nilai K
def evaluate_k(k):
    correct = 0
    for i in range(len(data)):
        test = data[i]
        train = data[:i] + data[i+1:]

        # hitung jarak ke semua data training
        dists = []
        for row in train:
            dist = math.sqrt(
                (test["Temperatur (°C)"] - row["Temperatur (°C)"])**2
+
                (test["Kecepatan Angin (km/jam)"] - row["Kecepatan
Angin (km/jam)"])**2)
            dists.append((dist, row["Persepsi"]))

        dists.sort(key=lambda x: x[0])
        k_neighbors = [label for _, label in dists[:k]]
        pred = Counter(k_neighbors).most_common(1)[0][0]

        if pred == test["Persepsi"]:
            correct += 1

    return correct / len(data)
```

```
print("Evaluasi K")
for k in range(1, 8):
    acc = evaluate_k(k)
    print(f"k = {k} → Akurasi = {acc*100:.2f}%")
```

```
Evaluasi K
k = 1 → Akurasi = 100.00%
k = 2 → Akurasi = 100.00%
k = 3 → Akurasi = 75.00%
k = 4 → Akurasi = 87.50%
k = 5 → Akurasi = 75.00%
k = 6 → Akurasi = 75.00%
k = 7 → Akurasi = 0.00%
```

Latihan 2

```
# Membuat Dataset
data = [
    {"NIM": "TI001", "Hasil Sebenarnya": "Lulus",       "Hasil
Prediksi": "Lulus"},
    {"NIM": "TI002", "Hasil Sebenarnya": "Lulus",       "Hasil
Prediksi": "Lulus"},
    {"NIM": "TI003", "Hasil Sebenarnya": "Lulus",       "Hasil
Prediksi": "Lulus"},
    {"NIM": "TI004", "Hasil Sebenarnya": "Lulus",       "Hasil
Prediksi": "Tidak Lulus"},
    {"NIM": "TI005", "Hasil Sebenarnya": "Lulus",       "Hasil
Prediksi": "Tidak Lulus"},
    {"NIM": "TI006", "Hasil Sebenarnya": "Tidak Lulus", "Hasil
Prediksi": "Lulus"},
    {"NIM": "TI007", "Hasil Sebenarnya": "Tidak Lulus", "Hasil
Prediksi": "Tidak Lulus"},
    {"NIM": "TI008", "Hasil Sebenarnya": "Tidak Lulus", "Hasil
Prediksi": "Tidak Lulus"},
    {"NIM": "TI009", "Hasil Sebenarnya": "Tidak Lulus", "Hasil
Prediksi": "Tidak Lulus"},
    {"NIM": "TI010", "Hasil Sebenarnya": "Tidak Lulus", "Hasil
Prediksi": "Tidak Lulus"},
]

df = pd.DataFrame(data)
print("DATA LATIHAN 2")
print(df)
```

```
DATA LATIHAN 2
     NIM Hasil Sebenarnya Hasil Prediksi
0  TI001            Lulus          Lulus
1  TI002            Lulus          Lulus
```

```
2  TI003            Lulus          Lulus
3  TI004            Lulus    Tidak Lulus
4  TI005            Lulus    Tidak Lulus
5  TI006      Tidak Lulus          Lulus
6  TI007      Tidak Lulus    Tidak Lulus
7  TI008      Tidak Lulus    Tidak Lulus
8  TI009      Tidak Lulus    Tidak Lulus
9  TI010      Tidak Lulus    Tidak Lulus
```

```
# Coenfusion Matrix
TP = sum((df["Hasil Sebenarnya"] == "Lulus") & (df["Hasil Prediksi"]
== "Lulus"))
FN = sum((df["Hasil Sebenarnya"] == "Lulus") & (df["Hasil Prediksi"]
== "Tidak Lulus"))
FP = sum((df["Hasil Sebenarnya"] == "Tidak Lulus") & (df["Hasil
Prediksi"] == "Lulus"))
TN = sum((df["Hasil Sebenarnya"] == "Tidak Lulus") & (df["Hasil
Prediksi"] == "Tidak Lulus"))

print("\n=== CONFUSION MATRIX ===")
print(pd.DataFrame({
    "Pred: Lulus": [TP, FP],
    "Pred: Tidak Lulus": [FN, TN]
}, index=["Actual: Lulus", "Actual: Tidak Lulus"]))
```

```
=== CONFUSION MATRIX ===
                    Pred: Lulus  Pred: Tidak Lulus
Actual: Lulus                 3                 2
Actual: Tidak Lulus           1                 4
```

```
# Acruacy preciesion recall
accuracy = (TP + TN) / (TP + TN + FP + FN)
precision = TP / (TP + FP)
recall = TP / (TP + FN)
print("\n=== METRIK ===")
print(f"Accuracy : {accuracy*100:.2f}%")
print(f"Precision: {precision*100:.2f}%")
print(f"Recall   : {recall*100:.2f}%")
```

```
=== METRIK ===
Accuracy : 70.00%
Precision: 75.00%
Recall   : 60.00%
```