In [1]:
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from collections import Counter
```

In [2]:
```
iris = load_iris()
X = iris.data
y = iris.target
```

In [3]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta
```

In [4]:
```
def euclidean_distance(x1, x2):
    return np.sqrt(np.sum((x1 - x2) ** 2))
```

In [5]:
```
def weighted_knn_predict(X_train, y_train, x_test, k=3):
    distances = []
    for i, x_train in enumerate(X_train):
        dist = euclidean_distance(x_train, x_test)
        distances.append((dist, y_train[i]))

    distances.sort(key=lambda x: x[0])

    k_neighbors = distances[:k]

    class_votes = {}
    for dist, label in k_neighbors:
        weight = 1 / (dist + 1e-5)
        class_votes[label] = class_votes.get(label, 0) + weight

    return max(class_votes, key=class_votes.get)
```

In [8]:
```
correct = 0
for i, x_test in enumerate(X_test):
    prediction = weighted_knn_predict(X_train, y_train, x_test, k=3)
    if prediction == y_test[i]:
        correct += 1

accuracy = correct / len(X_test)
print("Weighted KNN Accuracy:", accuracy)

print("\nSample Predictions:")
for i in range(5):
    pred = weighted_knn_predict(X_train, y_train, X_test[i], k=3)
    print(f"Test sample {i+1}: Predicted = {iris.target_names[pred]}, Actual = {iri
```

```
Weighted KNN Accuracy: 1.0

Sample Predictions:
Test sample 1: Predicted = versicolor, Actual = versicolor
Test sample 2: Predicted = setosa, Actual = setosa
Test sample 3: Predicted = virginica, Actual = virginica
Test sample 4: Predicted = versicolor, Actual = versicolor
Test sample 5: Predicted = versicolor, Actual = versicolor
```

In [ ]:
```
3
```