**5.**Develop a program to implement k-Nearest Neighbour algorithm to classify the randomly generated 100 values of x in the range of [0,1]. Perform the following based on dataset generated.  
**a)**Label the first 50 points {x1,……,x50} as follows: if (xi ≤ 0.5), then xi ∊ Class1, else xi ∊ Class1  
**b)** Classify the remaining points, x51,……,x100 using KNN. Perform this for k=1,2,3,4,5,20,30

import numpy as np

import matplotlib.pyplot as plt

from collections import Counter

# Generate random data

data = np.random.rand(100)

# Assign labels to the first 50 points

labels = ["Class1" if x <= 0.5 else "Class2" for x in data[:50]]

def euclidean\_distance(x1, x2):

return abs(x1 - x2)

def knn\_classifier(train\_data, train\_labels, test\_point, k):

distances = [(euclidean\_distance(test\_point, train\_data[i]), train\_labels[i]) for i in range(len(train\_data))]

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

k\_nearest\_neighbors = distances[:k]

k\_nearest\_labels = [label for \_, label in k\_nearest\_neighbors]

return Counter(k\_nearest\_labels).most\_common(1)[0][0]

# Split into training and testing sets

train\_data = data[:50]

train\_labels = labels

test\_data = data[50:]

# Define k values

k\_values = [1, 2, 3, 4, 5, 20, 30]

# Run k-NN classification

print("--- k-Nearest Neighbors Classification ---")

print("Training dataset: First 50 points labeled based on the rule (x <= 0.5 -> Class1, x > 0.5 -> Class2)")

print("Testing dataset: Remaining 50 points to be classified\n")

results = {}

for k in k\_values:

print(f"Results for k = {k}:")

classified\_labels = [knn\_classifier(train\_data, train\_labels, test\_point, k) for test\_point in test\_data]

results[k] = classified\_labels

for i, label in enumerate(classified\_labels, start=51):

print(f"Point x{i} (value: {test\_data[i - 51]:.4f}) is classified as {label}")

print("\n")

# Visualization

print("Classification complete.\n")

for k in k\_values:

classified\_labels = results[k]

class1\_points = [test\_data[i] for i in range(len(test\_data)) if classified\_labels[i] == "Class1"]

class2\_points = [test\_data[i] for i in range(len(test\_data)) if classified\_labels[i] == "Class2"]

plt.figure(figsize=(10, 6))

plt.scatter(train\_data, [0] \* len(train\_data), c=["blue" if label == "Class1" else "red" for label in train\_labels],

label="Training Data", marker="o")

plt.scatter(class1\_points, [1] \* len(class1\_points), c="blue", label="Class1 (Test)", marker="x")

plt.scatter(class2\_points, [1] \* len(class2\_points), c="red", label="Class2 (Test)", marker="x")

plt.title(f"k-NN Classification Results for k = {k}")

plt.xlabel("Data Points")

plt.ylabel("Classification Level")

plt.legend()

plt.grid(True)

plt.show()