In [ ]:

```
from math import sqrt
# calculate the Euclidean distance between two vectors
#     Euclidean Distance = sqrt(sum i to N (x1_i – x2_i)^2)
# Result:
#    10.295630140987
#    10.392304845413264
#          10.723805294763608
#          10.04987562112089
#        2.449489742783178
#        2.6457513110645907
#          3.1622776601683795
#          2.6457513110645907
def euclidean_distance(row1, row2):
        distance = 0.0
        for i in range(len(row1)-1):
                distance += (row1[i] - row2[i])**2
        return sqrt(distance)

# Locate the most similar neighbors
# Result
#   [6,5,7,5,6,7,1],
#   [5,6,6,6,5,7,1],
#       [7,6,7,6,5,6,1]]
def get_neighbors(train, test_row, num_neighbors):
        distances = list()
        for train_row in train:
                dist = euclidean_distance(test_row, train_row)
                distances.append((train_row, dist))
        distances.sort(key=lambda tup: tup[1])
        neighbors = list()
        for i in range(num_neighbors):
                neighbors.append(distances[i][0])
        return neighbors

# Make a classification prediction with neighbors
# - test_row is row 0
# - num_neighbors is 3
def predict_classification(train, test_row, num_neighbors):
        neighbors = get_neighbors(train, test_row, num_neighbors)
        output_values = [row[-1] for row in neighbors]
        prediction = max(set(output_values), key=output_values.count)
        return prediction

# Test distance function
dataset =  [[7,6,5,5,6,7,1],
            [1,2,3,2,1,3,0],
                  [2,1,3,3,1,2,0],
            [1,1,2,3,2,2,0],
                  [2,2,3,3,2,1,0],
                  [6,5,7,5,6,7,1],
                  [5,6,6,6,5,7,1],
                  [5,6,7,5,7,6,1],
                  [7,6,7,6,5,6,1]]

# Caluclate euclidean_distance
print("Euclidean distance between two vectors")
for i in range(1,9):
        print(euclidean_distance(dataset[0],dataset[i]))
```

```python
# row 0 (i.e., dataset[0]) is the one to be predicted
prediction = predict_classification(dataset, dataset[0], 3)

# - dataset[0][-1] is the last element of row 0 of dataset
# - Display
#    Expected 1, Got 1.
print('Expected %d, Got %d.' % (dataset[0][-1], prediction))
```

```
Euclidean distance between two vectors
10.295630140987
10.392304845413264
10.723805294763608
10.04987562112089
2.449489742783178
2.6457513110645907
3.1622776601683795
2.6457513110645907
Expected 1, Got 1.
```