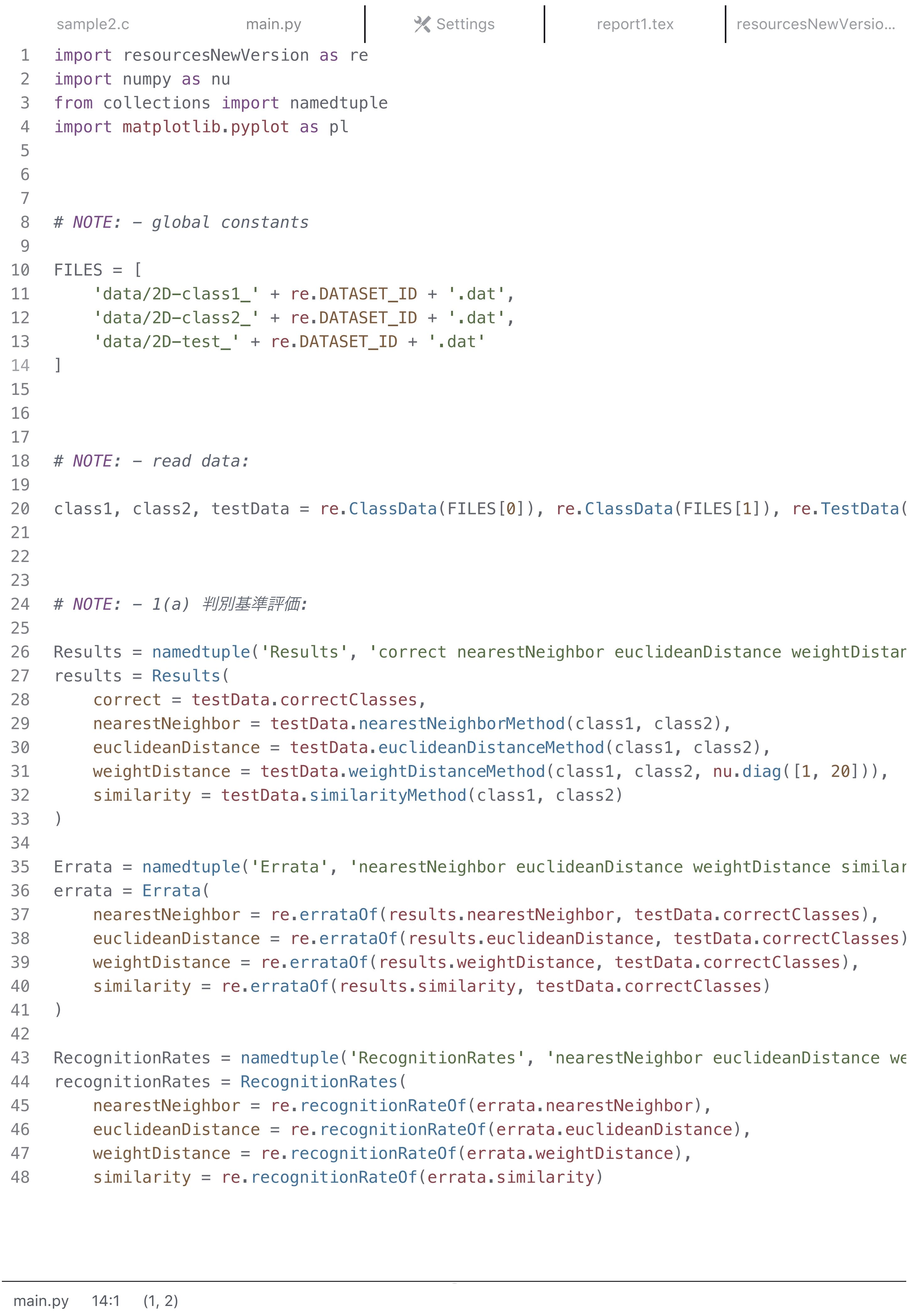

```python
import resourcesNewVersion as re
import numpy as nu
from collections import namedtuple
import matplotlib.pyplot as pl



# NOTE: - global constants

FILES = [
    'data/2D-class1_' + re.DATASET_ID + '.dat',
    'data/2D-class2_' + re.DATASET_ID + '.dat',
    'data/2D-test_' + re.DATASET_ID + '.dat'
]



# NOTE: - read data:

class1, class2, testData = re.ClassData(FILES[0]), re.ClassData(FILES[1]), re.TestData(



# NOTE: - 1(a) 判別基準評価:

Results = namedtuple('Results', 'correct nearestNeighbor euclideanDistance weightDistar
results = Results(
    correct = testData.correctClasses,
    nearestNeighbor = testData.nearestNeighborMethod(class1, class2),
    euclideanDistance = testData.euclideanDistanceMethod(class1, class2),
    weightDistance = testData.weightDistanceMethod(class1, class2, nu.diag([1, 20])),
    similarity = testData.similarityMethod(class1, class2)
)

Errata = namedtuple('Errata', 'nearestNeighbor euclideanDistance weightDistance similar
errata = Errata(
    nearestNeighbor = re.errataOf(results.nearestNeighbor, testData.correctClasses),
    euclideanDistance = re.errataOf(results.euclideanDistance, testData.correctClasses)
    weightDistance = re.errataOf(results.weightDistance, testData.correctClasses),
    similarity = re.errataOf(results.similarity, testData.correctClasses)
)

RecognitionRates = namedtuple('RecognitionRates', 'nearestNeighbor euclideanDistance we
recognitionRates = RecognitionRates(
    nearestNeighbor = re.recognitionRateOf(errata.nearestNeighbor),
    euclideanDistance = re.recognitionRateOf(errata.euclideanDistance),
    weightDistance = re.recognitionRateOf(errata.weightDistance),
    similarity = re.recognitionRateOf(errata.similarity)
```