1. 利用动态规划求解编辑距离问题

思路：当末尾字符相等时：c[m][n]=c[m-1][n-1]

当末尾字符不相等时，需要对两者之一的末尾进行编辑：

对字符串a或b的末尾进行修改, 以使之与b或a相等, 则此时c[m][n] = c[m - 1][n - 1] + 1

删除a末尾的元素,或者在b末尾添加元素, 使a与b相等, 则此时c[m][n] = c[m - 1][n] + 1;

删除b末尾的元素, 或者在b末尾添加元素, 使b与a相等, 则此时c[m][n] = c[m][n - 1] + 1;

源代码：

# -\*- coding: utf-8 -\*-

import numpy as np

def distance(a,b):

m = len(a)

n = len(b)

c = np.zeros((m+1,n+1))

for i in range(m+1):

c[i][0] = i

for j in range(n+1):

c[0][j] = j

for i in range(1,m+1):

for j in range(1,n+1):

if a[i-1] == b[j-1]:

delta = 0

else:

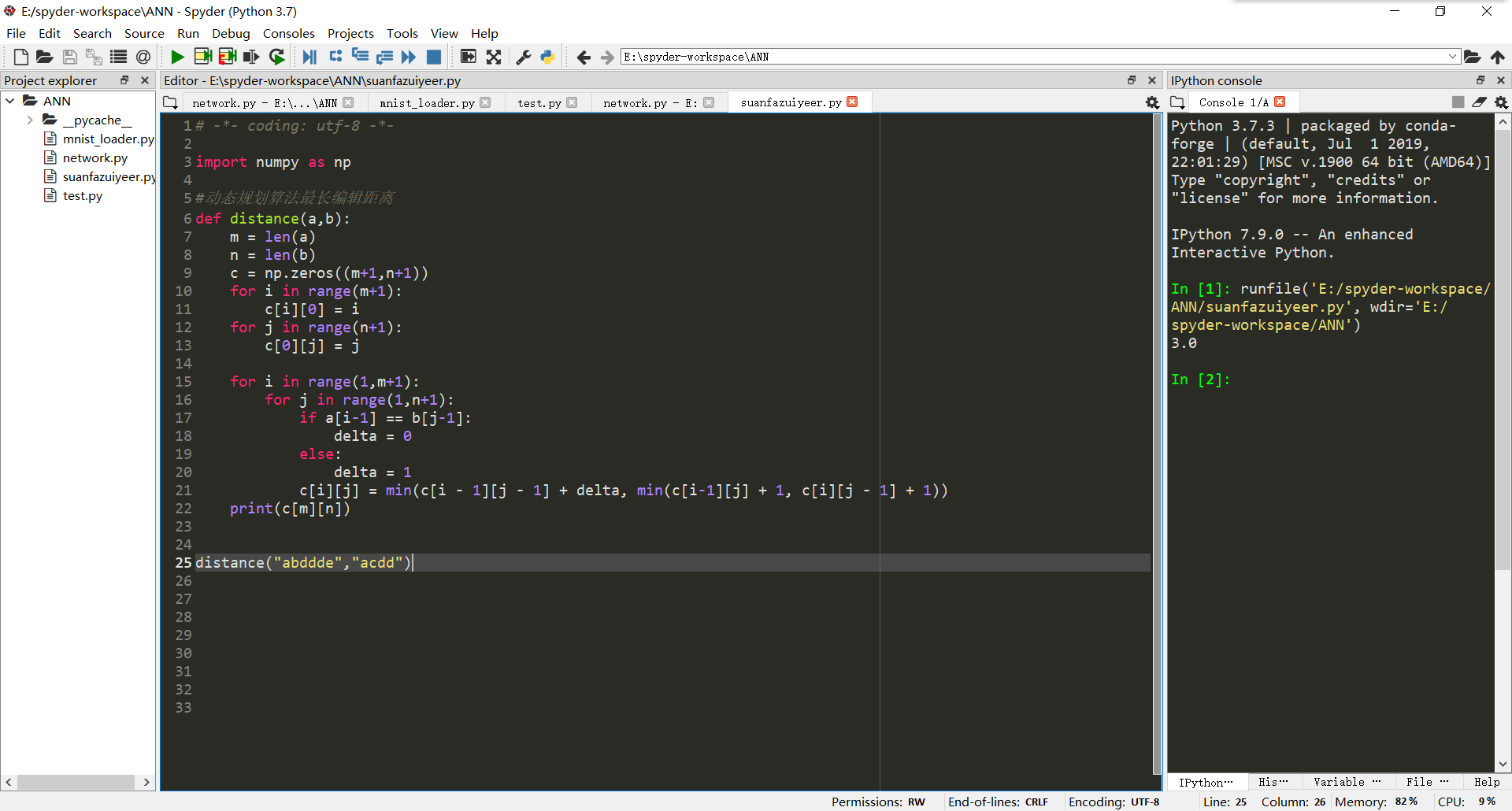
delta = 1

c[i][j] = min(c[i - 1][j - 1] + delta, min(c[i-1][j] + 1, c[i][j - 1] + 1))

print(c[m][n])

distance("abddde","acdd")

截图：



1. 利用动态规划算法求解最长公共子序列问题

思路：res[i][j]表示截止到字符串a的第i个字符和截止到字符串b的第j个字符的最长公共子序列

当i=0或者是j=0是，res[i][j]=0

当a[i]=b[j]时，res[i][j] = res[i-1][j-1]+1

当a[i]!=b[j]时，res[i][j]=max(res[i-1][j] res[i][j-1])

代码：

def LCS(a,b):

m = len(a)

n = len(b)

res = [[0 for i in range(m+1)] for j in range(n+1)]

for i in range(1,n+1):

for j in range(1,m+1):

if b[i-1] == a[j-1]:

res[i][j] = res[i-1][j-1]+1

else:

res[i][j] = max(res[i-1][j],res[i][j-1])

return res[-1][-1]

print(LCS("abcdef","fedcba"))

截图：

