[**https://github.com/Drozd363/ArcGIS\_arcpy\_S9\_15...git**](https://github.com/Drozd363/ArcGIS_arcpy_S9_15...git)

**Використання модуля NUMPY**

Ви будете писати програму modelMatrix(inMatrix, sequence) для розрахунку матриці трансформації. Наприклад, припустимо, що в просторі заданий вектор , яким буде результат застосування наступних трансформацій? (R(θ) означає обертання 2D на θ градусів і Т(х, у) означає переміщення 2D на осі х і у, відповідно, S(х, у) - масштабування).

Вигляд матриць трансформацій для окремих операцій наступний (для простору ):

Тобто результатом застосування виразів для вектору буде вираз:

inMatrix – словник, ключами якого є відповідні позначення для окремих матриць трансформацій (R – обертання, Т – переміщення, S – масштабування, V – вхідний вектор) наступного вигляду:

inMatrix = {'R':(33),'T':(1.0,2.0),'S':(2.0,1.2),'V':(1,2)}

sequence – рядок, в якому представлена послідовність застосування окремих матриць трансформацій, наприклад 'SRT' (для наведеного вище прикладу).

Функція повертає результат у вигляді списку значень

def modelMatrix(inMatrix, sequence):  
 *"""  
 Finds a transformation vector in 2D.  
  
 inMatrix: dictionary of transformation matrix  
 inMatrix = {'R':(angle),'T':( T\_x, T\_y),'S':( S\_x, S\_y),'V':(x,y)}  
 sequence: string of operation, like 'SRT' were T - first, S - last operation  
 returns: tuple of transformed vector  
 """* v = np.append(np.array(inMatrix['V']), 1)  
 v = np.matrix([v]).transpose()  
 rt = inMatrix['R']  
 st = inMatrix['S']  
 tt = inMatrix['T']  
 r = np.matrix([[np.cos(np.radians(rt)), -np.sin(np.radians(rt)), 0], [np.sin(np.radians(rt)), np.cos(np.radians(rt)), 0],  
 [0, 0, 1]])  
 s = np.matrix([[st[0], 0, 0], [0, st[1], 0], [0, 0, 1]])  
 t = np.matrix([[1, 0, tt[0]], [0, 1, tt[1]], [0, 0, 1]])  
 dictmatrix = {'R': r, 'T': t, 'S': s}  
 vect = 1  
 for i in sequence:  
 matr = dictmatrix[i]  
 vect = vect\*matr  
 outvector = np.ravel(vect\*v)  
 outtuple = (outvector[0], outvector[1], outvector[2])  
 return outtuple

>>> inMatrix = {'R':(33),'T':(1.0,2.0),'S':(2.0,1.2),'V':(1,2)}

>>> modelMatrix(inMatrix, 'SRT')

(-1.0, 5.33, 1)

>>> inMatrix = {'R':(16),'T':(1.0,2.0),'S':(1.0,2.0),'V':(1,2)}

>>> modelMatrix(inMatrix, 'TRS')

(0.86, 6.12, 1)

**Обов’язково** використайте numpy для матричних операцій.