Problema B. SemVerTia

Arquivo de entrada:	`standard input`
Arquivo de saída:	`standard output`
Limite de tempo:	1 segundo

Tia Marilda é uma professora entusiasta de Álgebra Linear e tem o hábito de ensinar seu sobrinho Jacinto sobre transformações geométricas aplicadas em imagens. Jacinto, porém, quando não entende uma transformação, reclama dizendo que ela "não tem **semvertia**". A Tia Marilda o corrige:

— "Jacinto, não é 'semvertia', é 'serventia'. Aplique corretamente as transformações!"

Jacinto está aprendendo duas transformações geométricas principais: escalamento e rotação. Agora, você deve ajudar Jacinto a aplicar corretamente essas transformações a uma matriz de uma imagem e verificar se ela "distorcida"ou, matematicamente, se ela é degenerada para uma dimensão inferior após as transformações.

Regras do problema:

Você receberá n transformações geométricas e deve aplicá-las sucessivamente a uma matriz identidade A de 2×2, que representa a imagem original. As transformações geométricas possíveis são:

1. **Rotação**: Representa a rotação de uma imagem por um ângulo θ em graus no sentido anti-horário. A matriz de rotação $R(\theta)$ é dada por:

$$R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

onde θ deve ser convertido de graus para radianos antes de aplicar as funções seno e cosseno.

2. **Escalamento**: Redimensiona a imagem ao longo dos eixos x e y. A matriz de escalamento é dada por:

$$S(sx, sy) = \begin{bmatrix} sx & 0 \\ 0 & sy \end{bmatrix}$$

Após aplicar todas as transformações, você deve calcular se a matriz resultante A tem seus vetores LI . Se a matriz não for invertível, a imagem "deformou", e você deve imprimir "SemVerTia". Caso contrário, imprima "SimTia, te vejo!".

Entrada

A primeira linha contém um inteiro n ($1 \leq n \leq 100$), o número de transformações geométricas.

As próximas n linhas contêm uma descrição das transformações. Cada transformação pode ser um dos seguintes tipos: - "R θ": onde θ é um número real representando o ângulo de rotação em graus. - "E sx sy": onde sx e sy são números reais que representam os fatores de escalamento para os eixos x e y.

Saída

Imprima "SemVerTia"se o determinante da matriz de transformação for 0. Caso contrário, imprima "SimTia, te vejo!".

Exemplos

standard input	standard output
2 R 90 E 1 1	SimTia, te vejo!

standard input	standard output
1 E 0 1	SemVerTia

Notas

No primeiro exemplo, aplicamos uma rotação de 90 graus e depois um escalamento uniforme nos dois eixos, resultando em uma transformação que preserva a "integridade"da imagem, então imprimimos "SimTia, te vejo!".

No segundo exemplo, a transformação de escalamento com fator 0 no eixo x faz com que a imagem "deforme", então imprimimos "SemVerTia".

Extra

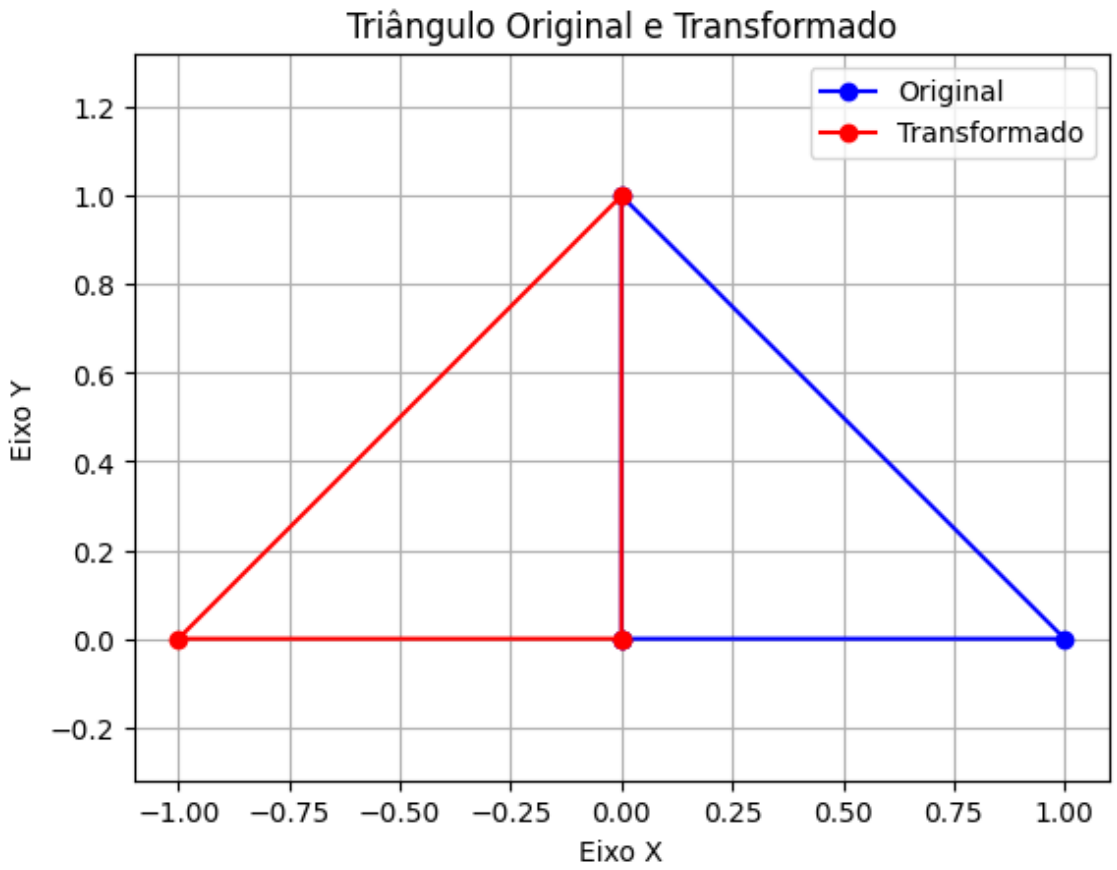
Além disso, sugerimos que você faça experimentos com outras matrizes que representem diferentes formas geométricas, como um triângulo, para verificar visualmente como as transformações afetam as imagens. Para isso, utilize a biblioteca Python ‘matplotlib‘ para plotar as formas geométricas antes e depois das transformações. Veja um exemplo:

- Escolha os vértices de um triângulo, por exemplo, $(0, 0)$, $(1, 0)$, e $(0, 1)$.
- Aplique uma rotação ou escalamento ao triângulo usando as transformações geométricas descritas.
- Plote o triângulo original e o transformado utilizando ‘matplotlib.pyplot‘ para visualizar o resultado.

Aqui está um exemplo de código em Python usando ‘matplotlib‘ para visualizar uma transformação de um triângulo:

```
import numpy as np
import matplotlib.pyplot as plt
triangulo = np.array([[0, 1, 0],
                      [0, 0, 1]])
theta = np.radians(90)
R = np.array([[np.cos(theta), -np.sin(theta)],
              [np.sin(theta), np.cos(theta)]])

triangulo_transformado = R @ triangulo
plt.figure()
plt.plot(np.append(triangulo[0, :], triangulo[0, 0]),
         np.append(triangulo[1, :], triangulo[1, 0]),
         'bo-', label='Original')
plt.plot(np.append(triangulo_transformado[0, :], triangulo_transformado[0, 0]),
         np.append(triangulo_transformado[1, :], triangulo_transformado[1, 0]),
         'ro-', label='Transformado')
plt.legend()
plt.axis('equal')
plt.grid()
plt.title('Triângulo Original e Transformado')
plt.xlabel('Eixo X')
plt.ylabel('Eixo Y')
plt.show()
```



Experimente criar outras figuras geométricas e aplicar diferentes transformações para observar como elas alteram o formato das imagens.

JLF