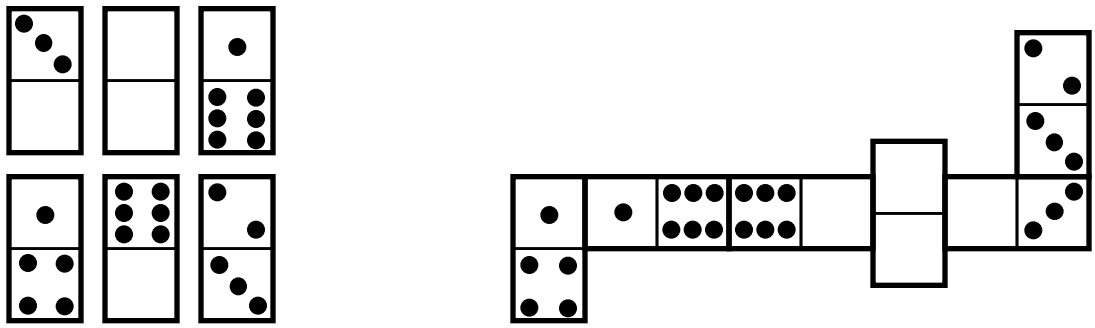
Dominó

Arquivo fonte: domino.c, domino.cc, domino.cpp ou domino.pas

Todos conhecem o jogo de dominós, em que peças com dois valores devem ser colocadas na mesa em seqüência, de tal forma que os valores de peças imediatamente vizinhas sejam iguais. O objetivo desta tarefa é determinar se é possível colocar todas as peças de um conjunto dado em uma formação válida.



(a) *(b)*

Conjunto de seis peças (a) e uma formação utilizando todas as seis peças (b)

1. Tarefa

É dado um conjuto de peças de dominó. Cada peça tem dois valores *X* e *Y*, com *X* e *Y* variando de 0 a 6 (*X* pode ser igual a *Y*). Sua tarefa é escrever um programa que determine se é possível organizar todas as peças recebidas em seqüência, obedecendo as regras do jogo de dominó.

2. Entrada

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de testes contém um número inteiro *N* que indica a quantidade de peças do conjunto. As *N* linhas seguintes contêm, cada uma, a descrição de uma peça. Uma peça é descrita por dois inteiros *X* e *Y* ($0 \leq X \leq 6$ e $0 \leq Y \leq 6$) que representam os valores de cada lado da peça. O final da entrada é indicado por $N = 0$.

Exemplo de Entrada

```
3
0 1
2 1
2 1
2
1 1
0 0
6
```

```
3 0
0 0
1 6
4 1
0 6
2 3
0
```

3. Saída

Para cada conjunto de teste da entrada seu programa deve produzir três linhas na saída. A primeira linha deve conter um identificador do conjunto de teste, no formato “Teste *n*”, onde *n* é numerado a partir de 1. A segunda linha deve conter a expressão “sim” se for possível organizar todas as peças em uma formação válida ou a expressão “nao” (note a ausência de acento) caso contrário. A terceira linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

Exemplo de Saída

```
Teste 1
sim

Teste 2
nao

Teste 3
sim
```

(esta saída corresponde ao exemplo de entrada acima)

4. Restrições

$0 \leq N \leq 100$ (N = 0 apenas para indicar o final da entrada)