ORDENADO, OU NÃO?

Um algoritmo de ordenação de vetores, como o da Figura 1, executa diversas permutas em posições do vetor até que o mesmo esteja completamente ordenado. Dessa forma, é possível saber se um vetor está ordenado, contando o número de permutas que o algoritmo de ordenação executa. Ou seja, em um vetor ordenado o algoritmo executa zero permutas, enquanto que em um vetor desordenado o algoritmo executa uma ou mais permutas.

```
for(j = 0; j < N-1; j++) { // Ordenação
    menor_i = j;
    for(i = j+1; i < N; i++)
        if(vetor[ i ] < vetor[menor_i])
            menor_i = i;
    aux = vetor[ j ];
    vetor[ j ] = vetor[menor_i];
    vetor[menor_i] = aux;
}
```

Figura 1 – Algoritmo de ordenação.

Faça um programa que leia um vetor de tamanho N ($2 \leq N \leq 50$) e, usando (modificando) o algoritmo de ordenação da Figura 1, diga se o vetor já estava ordenado e, caso contrário, diga que não está ordenado e quantas permutas P ($1 \leq P$) foram necessárias para ordenar o vetor.

Entrada

A entrada contém duas linhas: a primeira com um valor inteiro N, correspondente ao tamanho do vetor; e a segunda com N inteiros, correspondentes aos números que irão preencher o vetor.

Saída

Imprima a mensagem “esta ordenado: executou 0 permutas” ou “nao esta ordenado: executou P permutas”, de acordo com a especificação fornecida.

Exemplos

Entrada*	Saída
8 1 2 3 4 5 6 7 8	esta ordenado: executou 0 permutas
8 2 1 3 4 5 6 7 8	nao esta ordenado: executou 1 permutas
8 8 7 6 5 4 3 2 1	nao esta ordenado: executou 16 permutas
5 3 19 4 1 7	nao esta ordenado: executou 4 permutas
10 15 12 2 3 5 6 10 7 8 9	nao esta ordenado: executou 14 permutas
6 10 20 30 40 50 60	esta ordenado: executou 0 permutas

* Existe apenas 1 espaço entre os números da entrada. Na tabela acima, existem 2 espaços, apenas para facilitar a visualização.