C Planejamento de Tarefas

Time Limit: 1s

O gerente de uma equipe de logística estava tendo dificuldades com a execução das demandas, pois determinadas tarefas só poderiam ser executadas após o término de outras tarefas anteriores. Para mitigar o problema, ele contratou um estagiário de TI para escrever um software que, dadas as relações entre as N tarefas de sua equipe, gerasse diferentes sequências de execução da tarefa que respeitassem as relações de precedência entre elas. Em termos mais precisos, dadas N tarefas numeradas sequencialmente de 1 a N, deveria ser gerada uma sequência $a_1, a_2, \ldots, a_N$ tal que $1 \leq a_i \leq N$, $a_i \neq a_j$ para $i \neq j$ e, se a tarefa a_i deve ser executada antes da tarefa a_j, então $i < j$.

Contudo, após 6 meses de desenvolvimento, o estagiário comunicou ao gerente que o software ainda não estava pronto (pois ainda faltava a conclusão de um "componente importante") mas que já era possível determinar que não seria adequado utilizar o software para N maior ou igual a 10. O gerente ficou deveras desanimado, pois a empresa era grande e o número de tarefas que ele tinham em mente excedia este número em mais de 1000 vezes...

O estágio tentava resolver o problema por força bruta, gerando todas as $N!$ permutações possíveis das tarefas e depois identificando, uma a uma, quais permutações seriam sequências válidas. O "componente importante"que faltava era a justamente a função que classificaria as permutações como válidas ou não...

Sendo amigo do estagiário, e vendo uma oportunidade de mostrar ao seu amigo a ineficiência de sua abordagem (antes, claro, de auxiliá-lo a escrever o software que atendesse às demandas do cliente), escreva um software que, dada uma permutação P das N tarefas, determine primeira dentre as próximas M permutações que sucedem P na ordenação lexicográfica (P inclusive) que constitui uma sequência válida, quando for o caso.

Entrada

A entrada consiste em vários casos de teste. A primeira linha de um caso de teste contém o número N ($1 \leq N \leq 1.000$) de tarefas e o número M ($1 \leq M \leq 10.000$) de permutações consecutivas a serem consideradas. A segunda linha contém uma permutação P destes elementos, composta por N inteiros entre 1 e N, sem repetições e separados por espaços em branco.

A linha a seguir contém o número R ($1 \leq R \leq N(N-1)/2$) de relações de precedência entre as tarefas. As R linhas seguintes contém uma relação cada, na forma de um par de inteiros i e j ($1 \leq i, j \leq N, i \neq j$), que indica que a tarefa i deve ser finalizada antes da tarefa j.

A entrada termina quando $N = M = 0$, caso que não deve ser processado.

Saída

Para cada caso de teste deve ser impressa, em uma linha, a mensagem "Caso #t: Q permutações", onde t é o número do caso de teste (cuja contagem tem início com o número 1) e Q é o número de permutações que foram avaliadas (P inclusive) até que se tenha identificado uma sequência válida.

Na linha a seguir deve ser impressa a sequência identificada, onde os números deve ser separados por um espaço em branco (não há espaço em branco após o último número). Caso não exista tal sequência, deve ser impressa a mensagem "Todas invalidas".

Casos de teste consecutivos devem ser separados por uma linha em branco.

Exemplos de entradas	Exemplos de saídas
4 10 1 4 2 3 2 4 2 1 3 3 4 3 2 1 1 1 3 8 10000 2 1 3 4 5 6 7 8 7 7 8 1 6 3 7 4 6 2 5 5 8 1 2 5 18 2 1 3 5 4 1 3 2 0 0	Caso #1: 1 permutacoes 1 4 2 3 Caso #2: 2 permutacoes 1 2 3 Caso #3: 5041 permutacoes 3 1 2 4 5 6 7 8 Caso #4: 18 permutacoes Todas invalidas

Este problema foi elaborado para ensino e docência. Quaisquer coincidências com problemas já existentes favor entrar em contato (edsonalves@unb.br) para que as devidas providências sejam tomadas.