SPOJ TOPOSORT

Topological Sorting

Prof. Edson Alves

Faculdade UnB Gama

Sandro is a well organised person. Every day he makes a list of things which need to be done and enumerates them from 1 to n. However, some things need to be done before others. In this task you have to find out whether Sandro can solve all his duties and if so, print the correct order.

Sandro é uma pessoa muito organizada. A cada dia ele faz uma lista de coisas que ele precisa fazer e as enumera de 1 a n. Contudo, algumas coisas precisam ser feitas antes de outras. Neste problema você deve determinar se Sandro pode cumprir todas as suas tarefas e, em caso afirmativo, imprima-as na ordem correta.

Input

In the first line you are given an integer n and m $(1 \leq n \leq 10000,$ $1 \leq m \leq 1000000)$. On the next m lines there are two distinct integers x and y, $(1 \leq x, y \leq 10000)$ describing that job x needs to be done before job y.

Output

Print "`Sandro fails.`" if Sandro cannot complete all his duties on the list. If there is a solution print the correct ordering, the jobs to be done separated by a whitespace. If there are multiple solutions print the one, whose first number is smallest, if there are still multiple solutions, print the one whose second number is smallest, and so on.

Entrada

Na primeira linha há dois inteiros n e m $(1 \leq n \leq 10000, 1 \leq m \leq 1000000)$. Nas m linhas seguintes há dois inteiros distintos x e y $(1 \leq x, y \leq 10000)$, descrevendo que a tarefa x precisa ser cumprida antes da tarefa y.

Saída

Imprima "`Sandro fails.`" se Sandro não pode cumprir todas as tarefas da lista. Se há uma solução imprima-a na ordem correta, separando as tarefas por um espaço em branco. Se há múltiplas soluções imprima aquela cuja primeira tarefa tem o menor número. Se ainda restam múltiplas soluções, imprima a que tenha o segundo menor número, e assim por diante.

Exemplo de entrada e saída

Exemplo de entrada e saída

```
8 9
```

Exemplo de entrada e saída

8 9

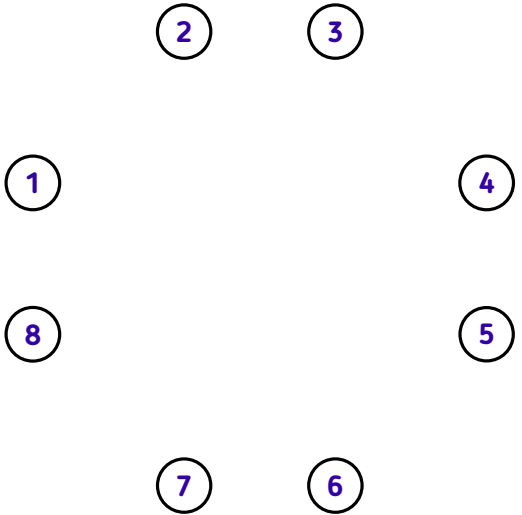
Exemplo de entrada e saída

8 9

 # de relações de dependência

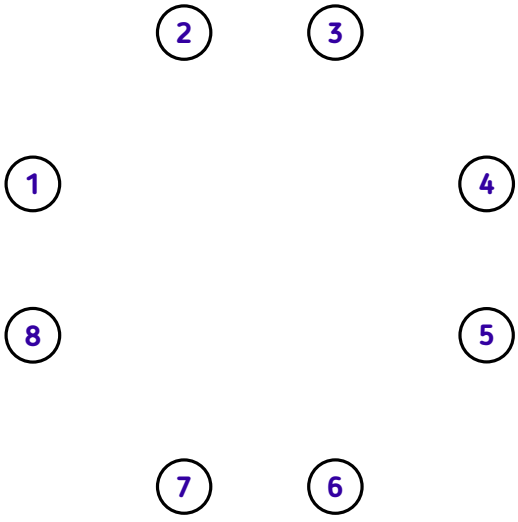
Exemplo de entrada e saída

```
8 9
```



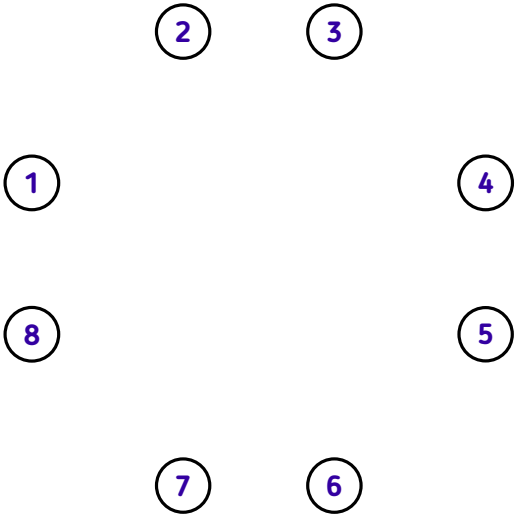
Exemplo de entrada e saída

```
8 9
1 4
```



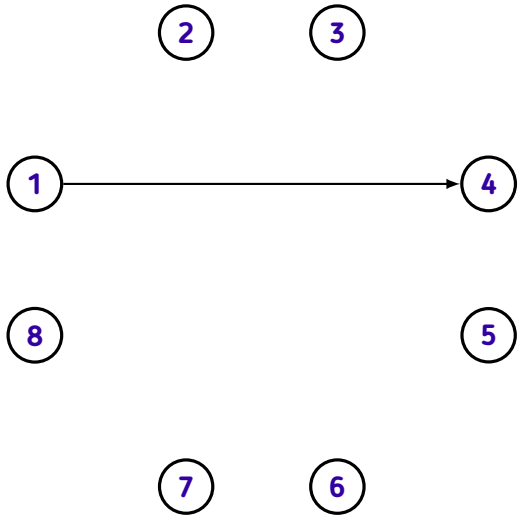
Exemplo de entrada e saída

8 9
1 4



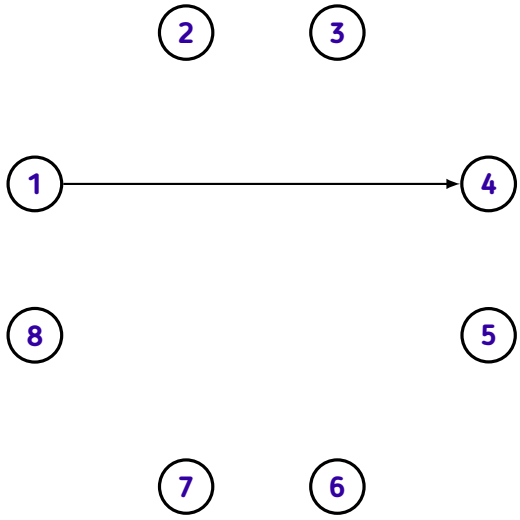
Exemplo de entrada e saída

```
8 9
1 4
```



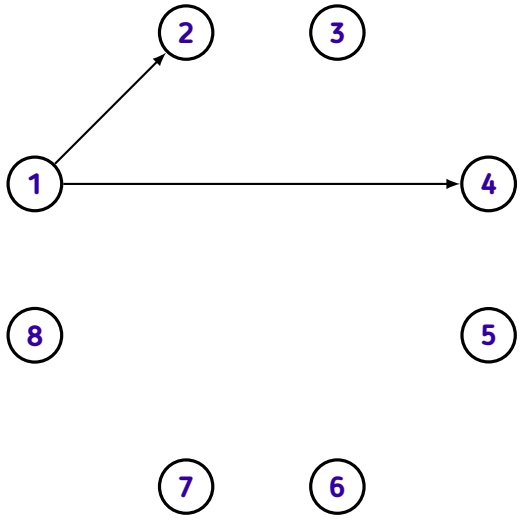
Exemplo de entrada e saída

```
8 9
1 4
1 2
```



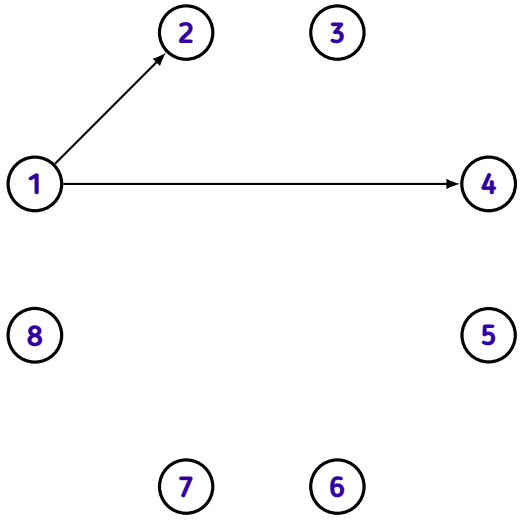
Exemplo de entrada e saída

```
8 9
1 4
1 2
```



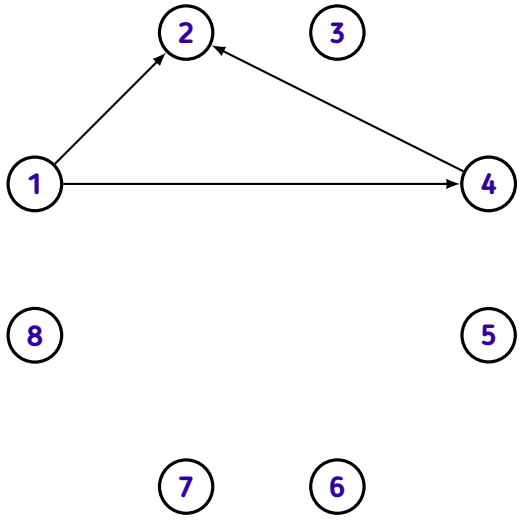
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
```



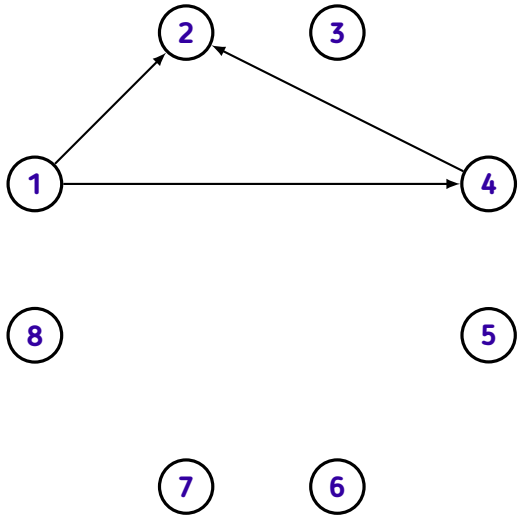
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
```



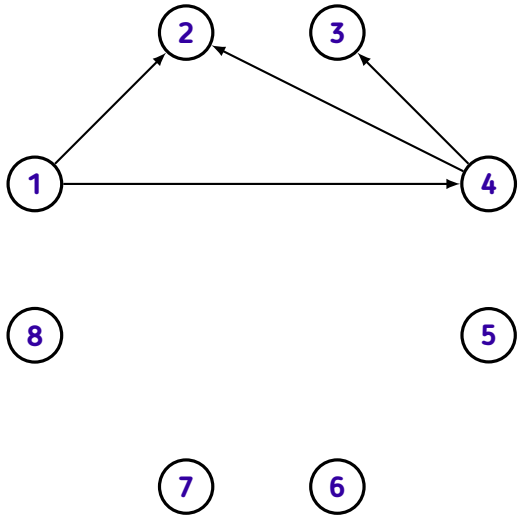
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
```



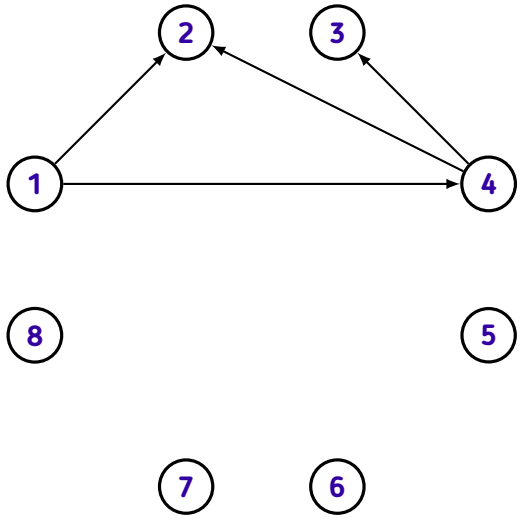
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
```



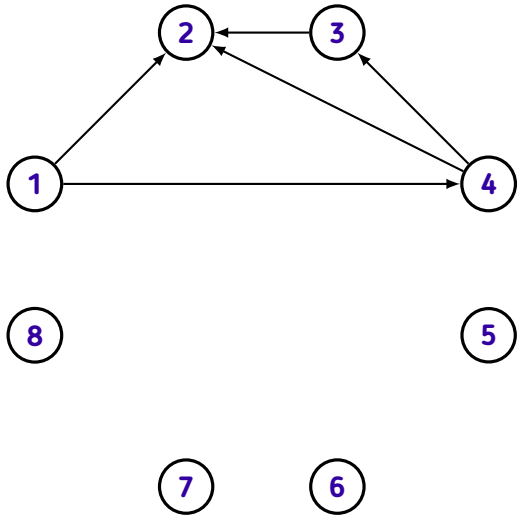
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
```



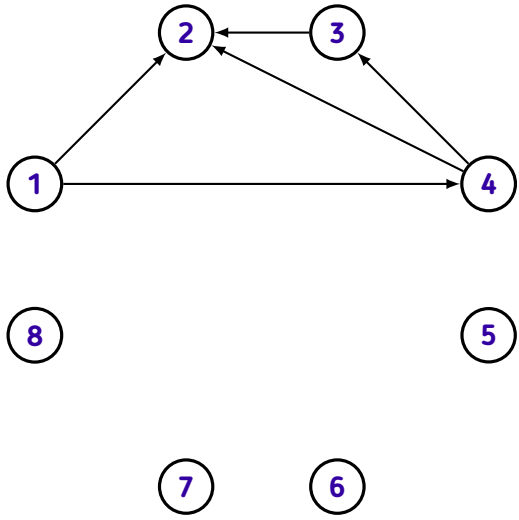
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
```



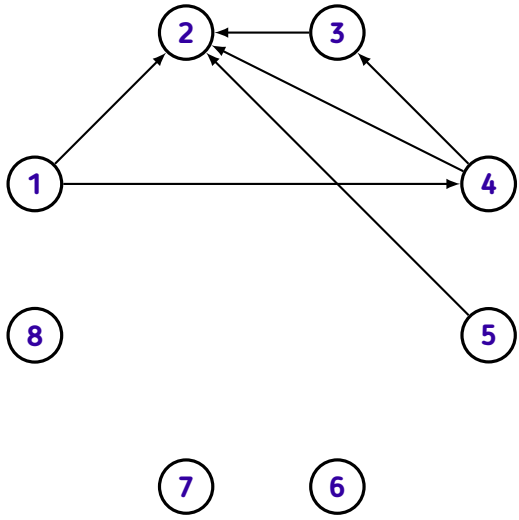
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
```



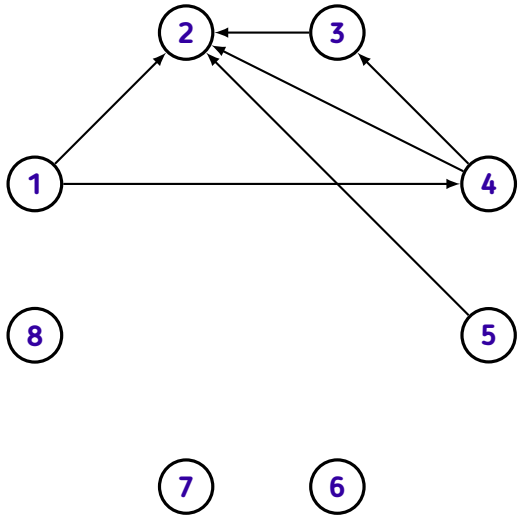
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
```



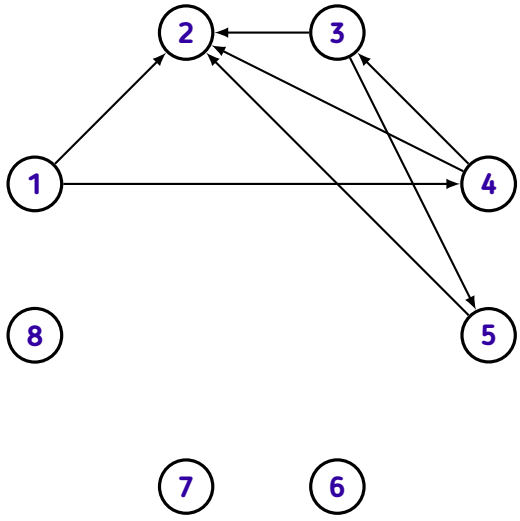
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
```



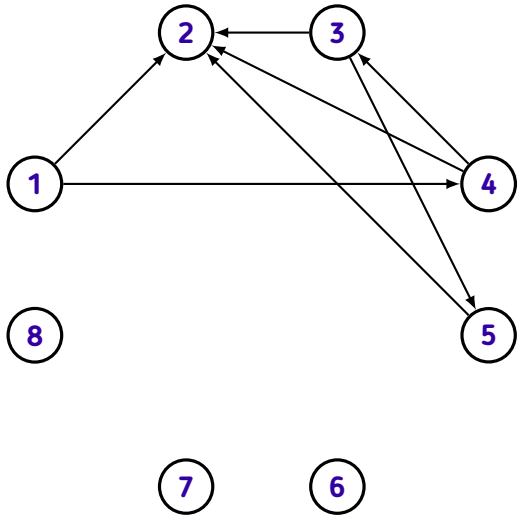
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
```



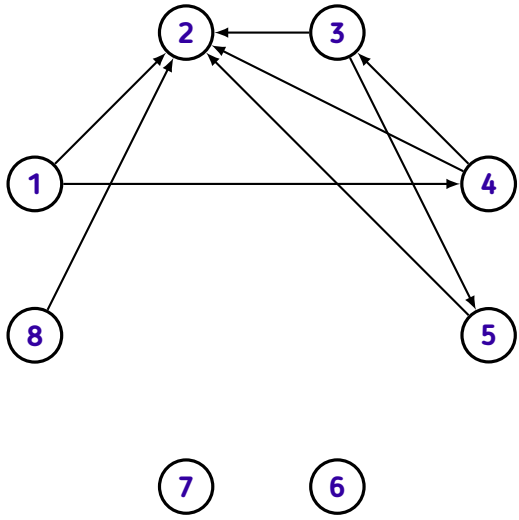
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
```



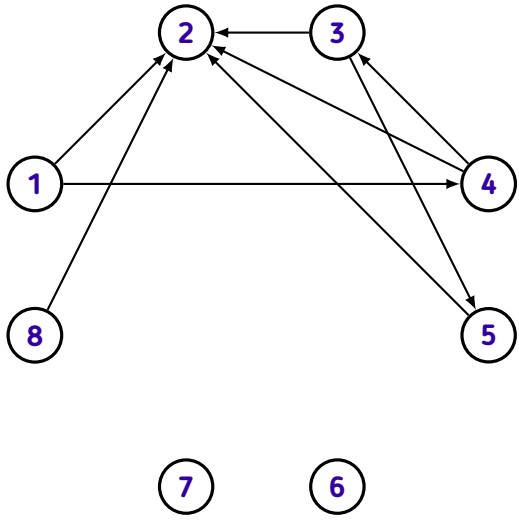
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
```



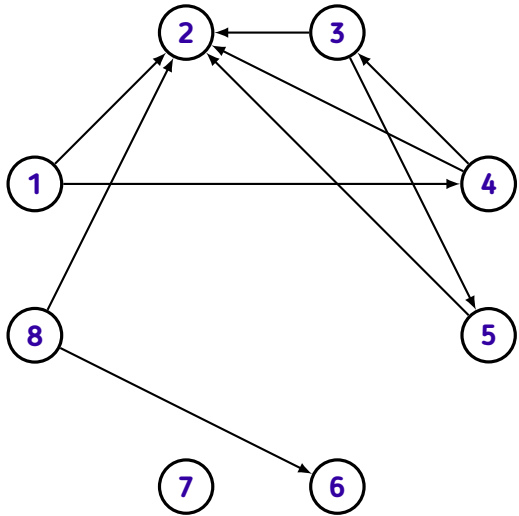
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



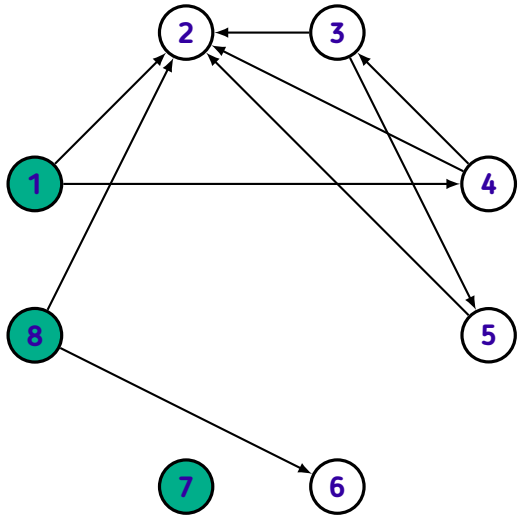
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



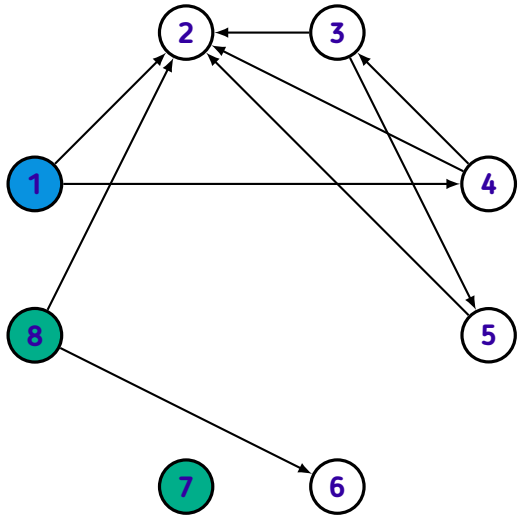
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



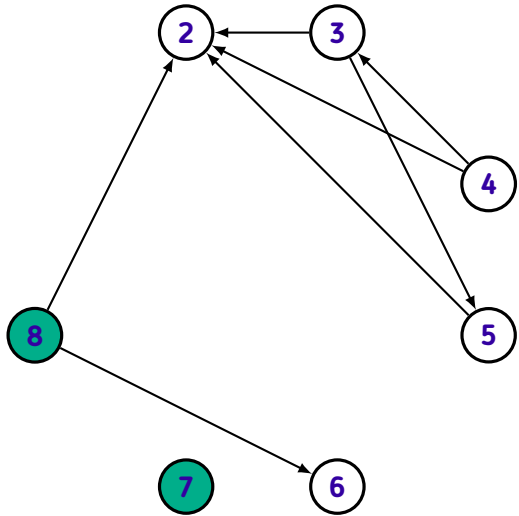
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



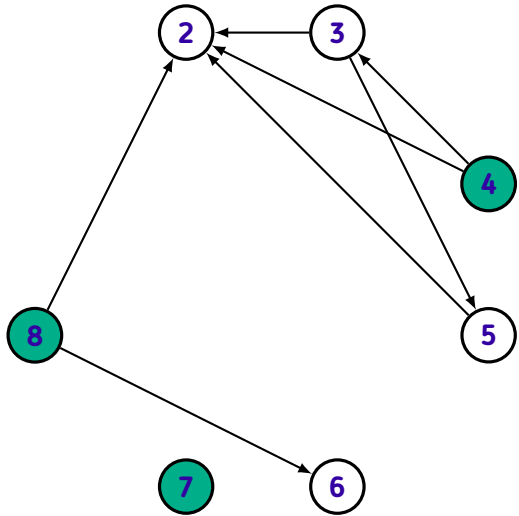
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



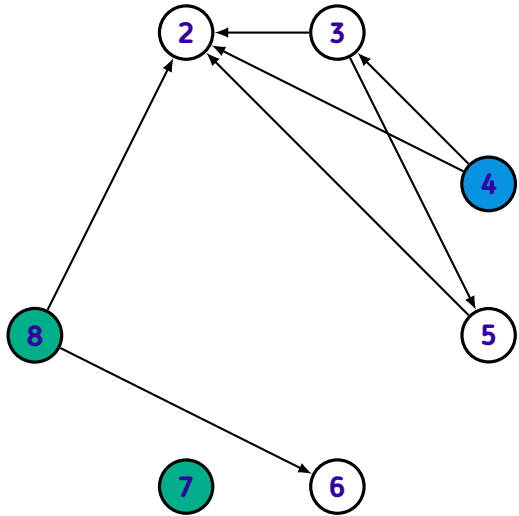
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



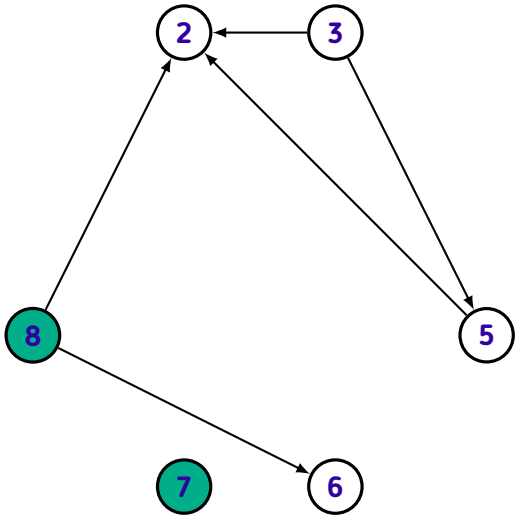
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



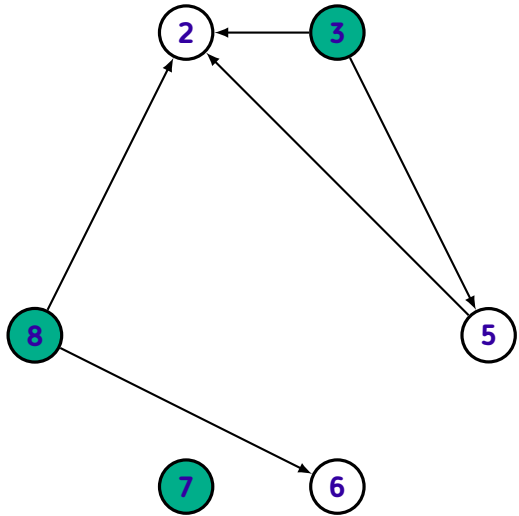
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



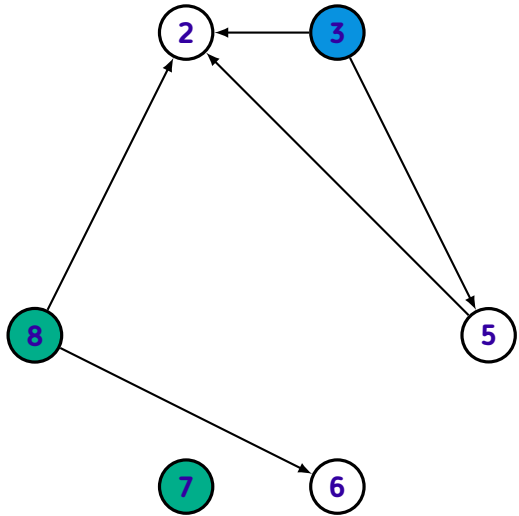
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



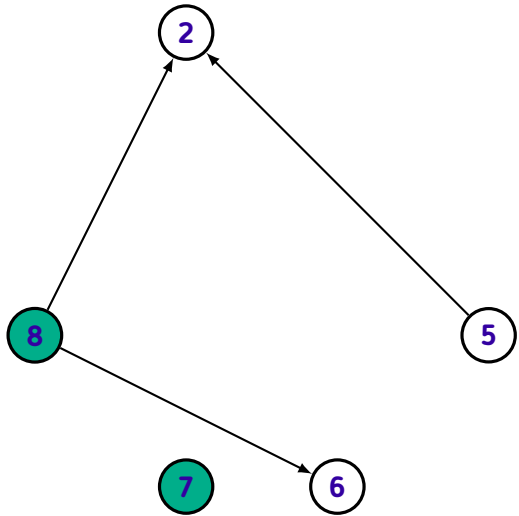
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



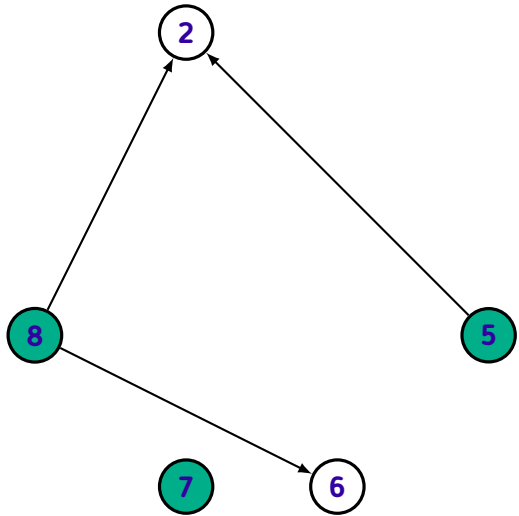
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



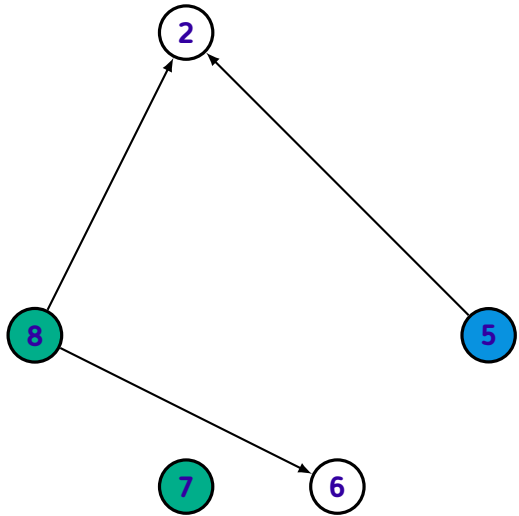
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



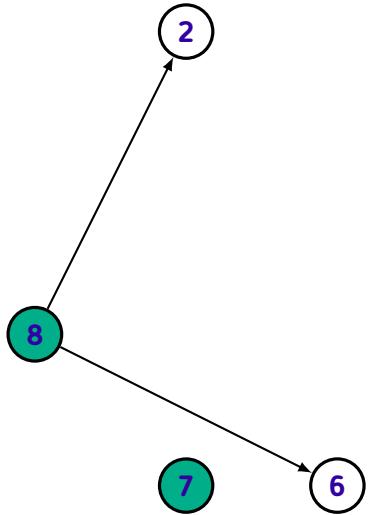
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



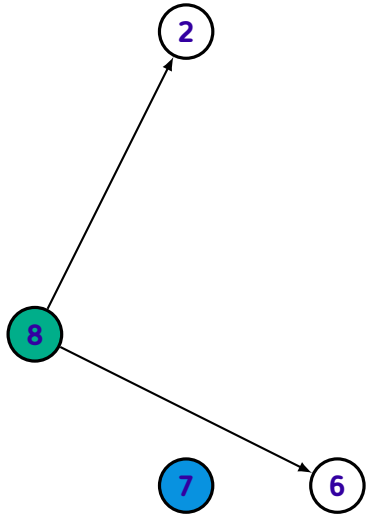
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



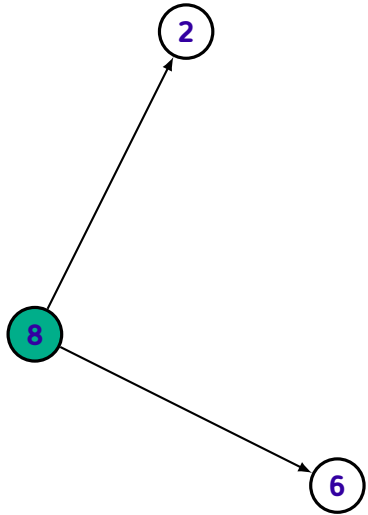
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



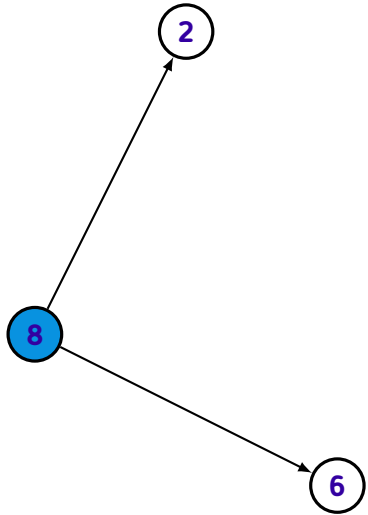
Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```



Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```

2

6

Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```

2

6

Exemplo de entrada e saída


```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```

Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```

Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```

Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```

Exemplo de entrada e saída

```
8 9
1 4
1 2
4 2
4 3
3 2
5 2
3 5
8 2
8 6
```

```
1 4 3 5 7 8 2 6
```

Exemplo de entrada e saída

Exemplo de entrada e saída

```
2 2
```

Exemplo de entrada e saída

```
2 2
```

Exemplo de entrada e saída

```
2 2
1 2
```

Exemplo de entrada e saída

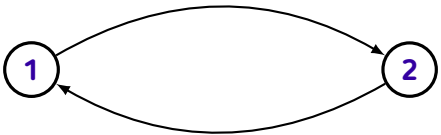
```
2 2
1 2
```

Exemplo de entrada e saída

```
2 2
1 2
2 1
```

Exemplo de entrada e saída

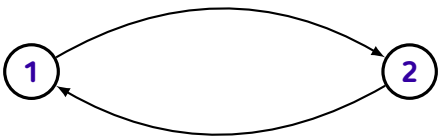
```
2 2
1 2
2 1
```



Exemplo de entrada e saída

```
2 2
1 2
2 1
```

```
Sandro fails.
```



Solução

Solução

- **Sandro será capaz de cumprir suas tarefas apenas se elas puderem ser ordenadas de tal forma que as prioridades sejam respeitadas**

Solução

- Sandro será capaz de cumprir suas tarefas apenas se elas puderem ser ordenadas de tal forma que as prioridades sejam respeitadas

- Em outras palavras, há solução apenas se existe uma ordenação topológica

Solução

- Sandro será capaz de cumprir suas tarefas apenas se elas puderem ser ordenadas de tal forma que as prioridades sejam respeitadas

- Em outras palavras, há solução apenas se existe uma ordenação topológica

- Se o grafo tem um ou mais ciclos, a resposta é `Sandro fails.`

Solução

- Sandro será capaz de cumprir suas tarefas apenas se elas puderem ser ordenadas de tal forma que as prioridades sejam respeitadas
- Em outras palavras, há solução apenas se existe uma ordenação topológica
- Se o grafo tem um ou mais ciclos, a resposta é `Sandro fails.`
- O problema pede, na saída, uma ordenação específica

Solução

- ⋆ **Sandro será capaz de cumprir suas tarefas apenas se elas puderem ser ordenadas de tal forma que as prioridades sejam respeitadas**
- ⋆ **Em outras palavras, há solução apenas se existe uma ordenação topológica**
- ⋆ **Se o grafo tem um ou mais ciclos, a resposta é `Sandro fails.`**
- ⋆ **O problema pede, na saída, uma ordenação específica**
- ⋆ **Esta ordenação pode ser obtida se a fila do algoritmo de Kahn for substituída por uma *min heap***

```
vector<int> topological_sort(int N)
{
    vector<int> o;
    priority_queue<int, vector<int>, greater<int>> q;

    for (int u = 1; u <= N; ++u)
        if (in[u].empty())
            q.push(u);

    while (not q.empty())
    {
        auto u = q.top();
        q.pop();

        o.emplace_back(u);

        for (auto v : out[u])
        {
            in[v].erase(u);
```

```
            if (in[v].empty())
                q.push(v);
        }
    }

    return (int) o.size() == N ? o : vector<int> { };
}

vector<int> solve(int N)
{
    return topological_sort(N);
}
```