

Atividade 2

Implementação com Lista de Adjacências

Especificação

Seu programa deve fazer a implementação, em linguagem C, de um TAD capaz de representar um **grafo não orientado** e **deve utilizar o conceito de LISTA de adjacências**. Além disso, seu programa **deve ser capaz de executar as seguintes operações**:

- **Criar grafo**: cria um grafo com um determinado número de vértices, onde **inicialmente não há arestas**.
- **Inserir aresta**: cria uma aresta entre dois vértices do grafo
- **Remover aresta**: remove uma aresta do grafo
- **Exibir lista de adjacências**: exibe na tela a lista de adjacências do grafo
- **Deletar grafo**: deleta o grafo liberando qualquer memória alocada por ele

O programa será executado e avaliado pelo run.codes:

- **Turma A (Terça)**: A6H7
- **Turma B (Segunda)**: JZW9

Para isso, deve respeitar a seguinte especificação de entrada e saída:

Entrada

A primeira linha da entrada contém dois inteiros n ($2 \leq n \leq 100$) e m ($0 \leq m \leq \frac{n(n-1)}{2}$) ; n representa o número de vértices no grafo que seu programa deverá criar; m representa o número de arestas que deverão ser inseridas inicialmente.

Em seguida, haverá m linhas contendo as arestas iniciais, cada linha m_i contém um par de inteiros u_i e v_i ($0 \leq u_i, v_i < n$ e $u_i \neq v_i$) , os vértices conectados pela aresta (**lembre-se que o primeiro vértice é o 0 e o último o n-1**).

Após isso, será recebido um inteiro q ($0 < q \leq 10$) , que indica o número de operações a serem executadas.

Segue-se na linha de baixo o primeiro valor q_1 ($1 \leq q_1 \leq 3$) indicando qual é a operação; A entrada necessária para a operação é dada logo em seguida, na mesma linha, baseada no valor de q_1 e pode ser:

- $q_1 = 1$ (**Inserir aresta**)

Um par de inteiros u_j e v_j ($0 \leq u_j, v_j < n$ e $u_j \neq v_j$), indicando que uma aresta deverá ser inserida entre os vértices u_j e v_j (se ainda não houver).

- $q_1 = 2$ (**Remover aresta**)

Um par de inteiros u_k e v_k ($0 \leq u_k, v_k < n$ e $u_k \neq v_k$), indicando que deverá ser removida a aresta entre os vértices u_k e v_k (se existir).

- $q_1 = 3$ (**Exibir lista de adjacências**)

A lista é apenas impressa na tela, não necessita de entrada adicional.

Esse processo é repetido para os demais valores de q_i .

Após isso a entrada é finalizada!

Saída

A única saída do seu programa é a saída da operação **Exibir lista de adjacências**:

Deve ser impressa uma linha na tela para cada vértice, esses são impressos em ordem crescente. Cada linha contém o número i do vértice em questão seguido por dois pontos e espaço (" i : "); em seguida, se houverem, são impressos os elementos da lista desse vértice separados por espaços (**Por segurança, garanta que haja um espaço entre o último valor de cada linha impressa e o caractere '\n' dessa linha! Também lembre que caso não haja elementos, imprime-se apenas um '\n' após " i : "**). Depois de impressa a última linha da lista, deve ser colocado um '\n' adicional.

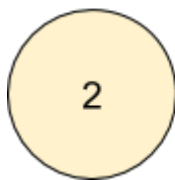
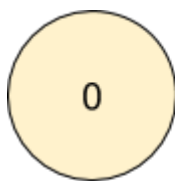
Exemplo

1. O programa recebe a primeira linha com os valores n e m :

3 2 \n

Criando o seguinte grafo:

Grafo:



Lista de adjacências:

0:

1:

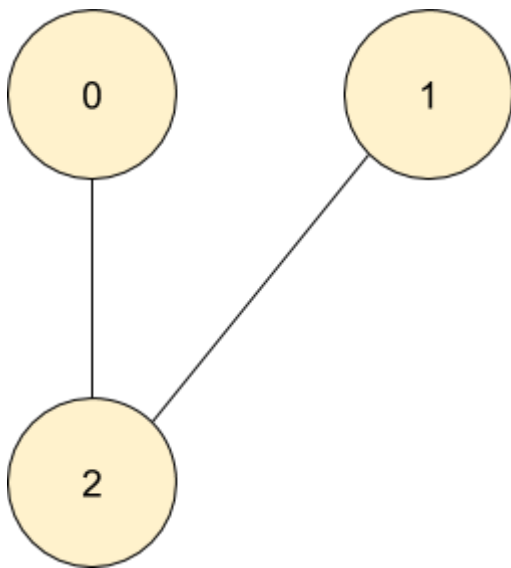
2:

2. Então recebe $m = 2$ linhas, contendo as arestas iniciais:

```
1 2\n0 2\n
```

Resultando em:

Grafo:



Lista de adjacências:

```
0: 2\n1: 2\n2: 1 0\n
```

3. O número $q = 3$ de operações então é passado:

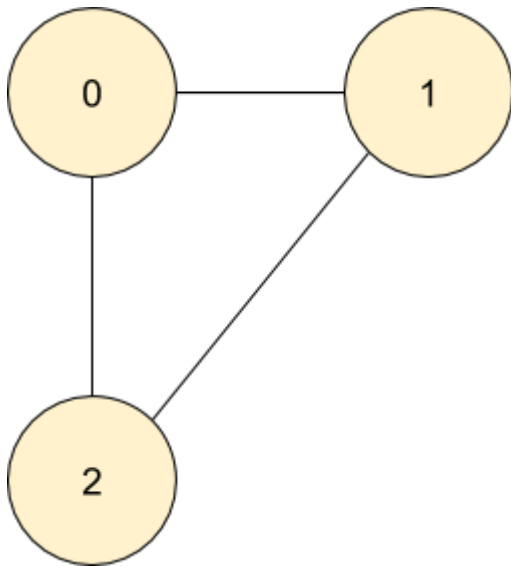
```
3\n
```

4. Em seguida, recebe a operação q_1 e sua respectiva entrada:

1 0 1 \n

Resultando em:

Grafo:



Matriz de adjacências:

0: 2 1

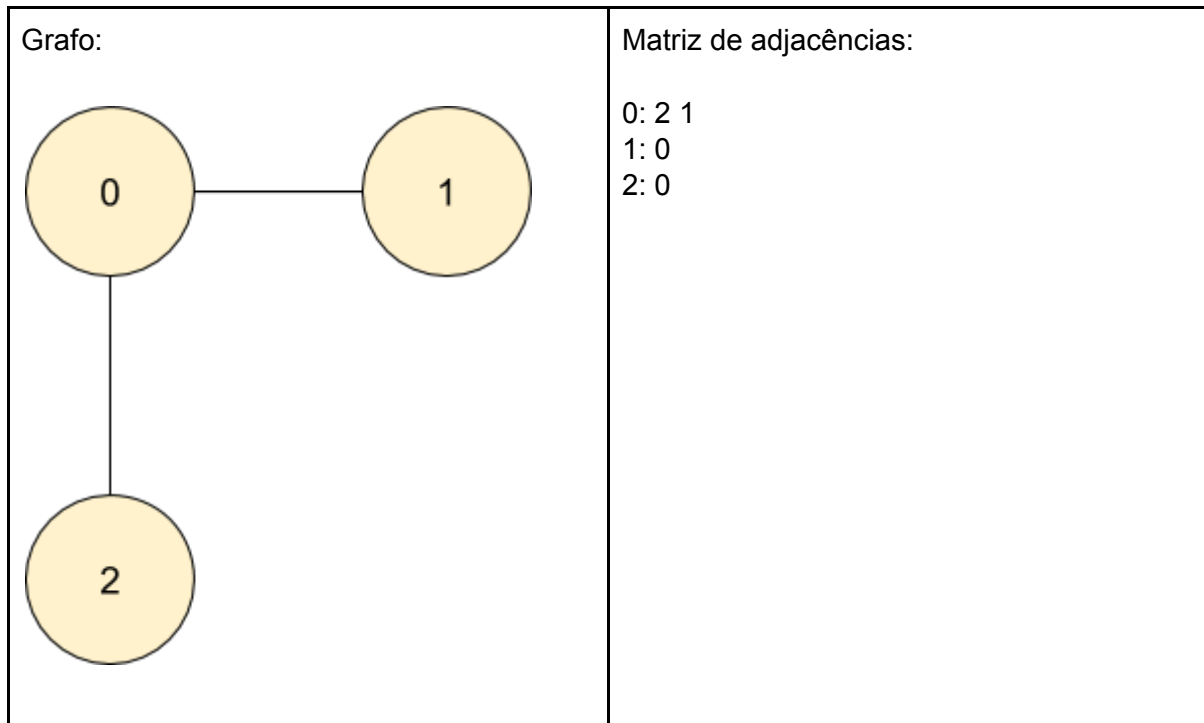
1: 2 0

2: 1 0

5. A operação q_2 e sua respectiva entrada é passada:

2·1·2\n

Resultando em:



6. Por fim, recebe a operação q_3 (a operação não necessita de mais parâmetros, pois $q_3 = 3$):

3\n

Fazendo com que a matriz de adjacência seja exibida na tela:

0:2·1·\n
1:0·\n
2:0·\n
\n

Fim da execução