Graf dirigit amb matriu d'adjacència. Quantes arestes diferents es poden visitar des de cada vèrtex — X94168_ca

Donada la classe *graf* que permet gestionar grafs dirigits i no etiquetats amb n vèrtexs (els vèrtexs són enters dins l'interval $[0, n-1]$), cal implementar el mètode

```
vector<nat> quantes_arestes_es_visiten () const;
// Pre: Cert
// Post: Retorna quantes arestes diferents es poden visitar (hi ha un camí)
// des de cada vèrtex del graf.
```

Les arestes es guarden en una matriu d'adjacència. Un dels jocs de prova públics és aquest graf que conté 5 vèrtexs (mira el PDF de l'enunciat):
el qual donaria com a resultat el vector 7 1 7 0 7, indicant que hi ha 7 arestes diferents que es poden visitar des del vèrtex 0 (totes les arestes), hi ha una des del vèrtex 1 (l'aresta que va de 1 a 3), hi ha 7 des del vèrtex 2 (totes), no n'hi ha cap des del vèrtex 3 i hi ha 7 des del vèrtex 4 (totes).
Cal enviar a jutge.org la següent especificació de la classe *graf* i la implementació del mètode dins del mateix fitxer (la resta de mètodes públics ja estan implementats). Indica dins d'un comentari a la capçalera del mètode el seu cost en funció del nombre de vèrtexs n i el nombre d'arestes m del graf.

```
#include <vector>
using namespace std;
typedef unsigned int nat;

class graf {
  // Graf dirigit i no etiquetat.
  // Les arestes es guarden en una matriu d'adjacència.
  public:
    // Constructora per defecte. Crea un graf buit.
    graf ();

    // Destructora
    ~graf ();

    // Llegeix les dades del graf del canal d'entrada
    void llegeix ();

    vector<nat> quantes_arestes_es_visiten () const;
    // Pre: Cert
    // Post: Retorna quantes arestes diferents es poden visitar (hi ha un camí)
    // des de cada vèrtex del graf.

  private:
    nat n; // Nombre de vèrtexs
    nat m; // Nombre d'arestes
    vector<vector<bool> > a; // Matriu d'adjacència
```

```
    // Aquí va l'especificació dels mètodes privats addicionals

};

// Aquí va la implementació del mètode públic quantes_arestes_es_visiten i privats addicionals
```

Degut a que jutge.org només permet l'enviament d'un fitxer amb la solució del problema, en el mateix fitxer hi ha d'haver l'especificació de la classe i la implementació del mètode *quantes_arestes_es_visiten* (el que normalment estarien separats en els fitxers *.hpp* i *.cpp*).
Per testejar la classe disposes d'un programa principal que llegeix un graf i després crida el mètode *quantes_arestes_es_visiten*.

Entrada

L'entrada conté un graf: el nombre de vèrtexs, el nombre d'arestes i una llista d'arestes. Cada aresta s'indica pels dos vèrtexs que relaciona.

Sortida

Escriu una línia amb el nombre d'arestes diferents que es poden visitar des de cada vèrtex del graf separats per espais.

Observació

Només cal enviar la classe requerida i la implementació del mètode *quantes_arestes_es_visiten*.
Pots ampliar la classe amb mètodes privats. Segueix estrictament la definició de la classe de l'enunciat.
Indica dins d'un comentari a la capçalera del mètode el seu cost en funció del nombre de vèrtexs n i el nombre d'arestes m del graf.

Exemple d'entrada 1

```
1
0
```

Exemple de sortida 1

```
0
```

Exemple d'entrada 2

```
2
0
```

Exemple de sortida 2

```
0 0
```

Exemple d'entrada 3

```
2
1
0 1
```

Exemple de sortida 3

```
1 0
```

Exemple d'entrada 4

```
2
2
0 1
1 0
```

Exemple de sortida 4

```
2 2
```

Exemple d'entrada 5

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

Exemple de sortida 5

```
4 4 4
```

Exemple d'entrada 6

```
5
7
4 0
0 2
0 1
2 1
2 4
2 3
1 3
```

Exemple de sortida 6

```
7 1 7 0 7
```

Exemple d'entrada 7

```
6
9
1 5
1 0
3 1
4 0
0 5
5 1
2 3
0 1
5 0
```

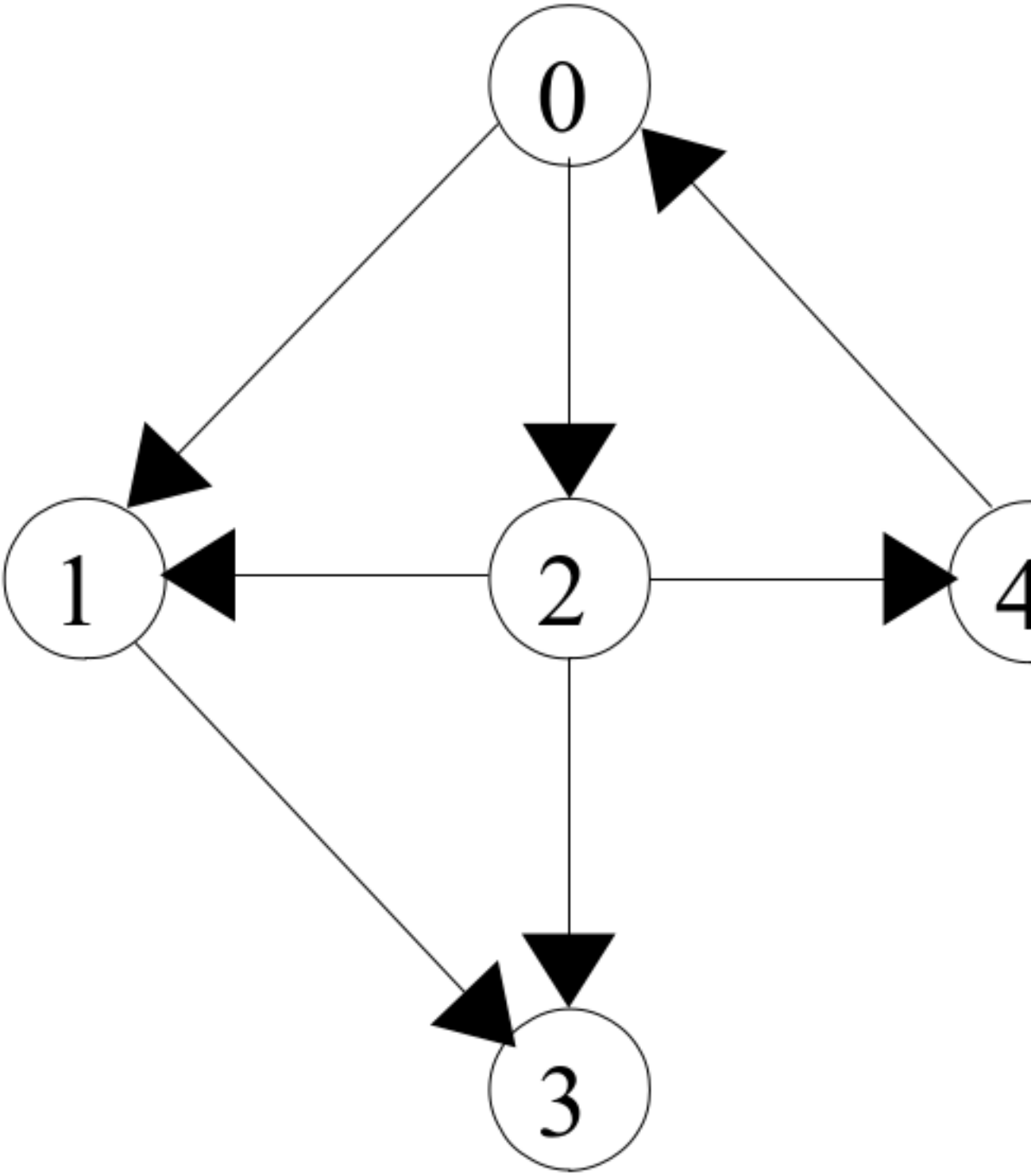
Exemple de sortida 7

```
6 6 8 7 7 6
```

Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T17:24:07.166Z


0
1
2
4
3