Graf dirigit amb llistes d'adjacència. Hi ha cicles des de cada vèrtex? Y89716_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<bool> hi_ha_cicles() const;
// Pre: Cert
// Post: Retorna si hi ha algun cicle des de cada vèrtex del graf.
```

Les arestes es guarden en llistes d'adjacència: un vector de n elements que conté llistes simplement encadenades amb els successors de cadascun dels n vèrtexs. Un dels jocs de prova públics és aquest graf que conté 5 vèrtexs (mira el PDF de l'enunciat):
les seves arestes estarien guardades en un vector amb 5 llistes d'adjacència, cada llista conté els successors de cadascun dels 5 vèrtexs:

```
0: 2, 1
1: 3
2: 1, 4, 3
3:
4: 0
```

el qual donaria com a resultat el vector T F T F T (T=true, F=false), indicant que hi ha cicles des dels vèrtexs 0, 2 i 4. En canvi no es detecten cicles des dels vèrtexs 1 i 3.
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 de vèrtexs [0, n-1]
  // Les arestes es guarden en llistes d'adjacència (llistes
  // simplement encadenades amb els successors de cada vèrtex).
  public:
    // Constructora per defecte. Crea un graf buit.
    graf();

    // Destructora
    ~graf();

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

    vector<bool> hi_ha_cicles() const;
    // Pre: Cert
```

```
    // Post: Retorna si hi ha algun cicle des de cada vèrtex del graf.

  private:
    nat n; // Nombre de vèrtexs
    nat m; // Nombre d'arestes

    struct node_succ {
      nat _succ;          // Vèrtex successor
      node_succ* _seg;    // Següent successor
    };
    vector<node_succ *> a; // Vector amb llistes simplement encadenades
                           // dels successors de cada vèrtex

    // Aquí va l'especificació dels mètodes privats addicionals

};

// Aquí va la implementació del mètode públic i dels mètodes 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 *hi_ha_cicles* (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 *hi_ha_cicles*.

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 un booleà per cada vèrtex (T=true, F=false) separats per espais que indiquen si hi ha algun cicle des de cada vèrtex del graf.

Observació

Només cal enviar la classe requerida i la implementació del mètode *hi_ha_cicles*. 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

```
F
```

Exemple d'entrada 2

```
2
0
```

Exemple de sortida 2

```
F F
```

Exemple d'entrada 3

```
2
1
0 1
```

Exemple de sortida 3

```
F F
```

Exemple d'entrada 4

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

Exemple de sortida 4

```
T T
```

Exemple d'entrada 5

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

Exemple de sortida 5

```
T T T
```

Exemple d'entrada 6

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

Exemple de sortida 6

```
T F T F T
```

Exemple d'entrada 7

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

Exemple de sortida 7

```
F F F F F
```

Exemple d'entrada 8

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

Exemple de sortida 8

```
T T T T T T
```

Exemple d'entrada 9

```
10
14
0 1
2 3
4 5
6 7
8 9
0 2
2 4
3 4
6 8
3 1
5 8
7 5
1 0
1 9
```

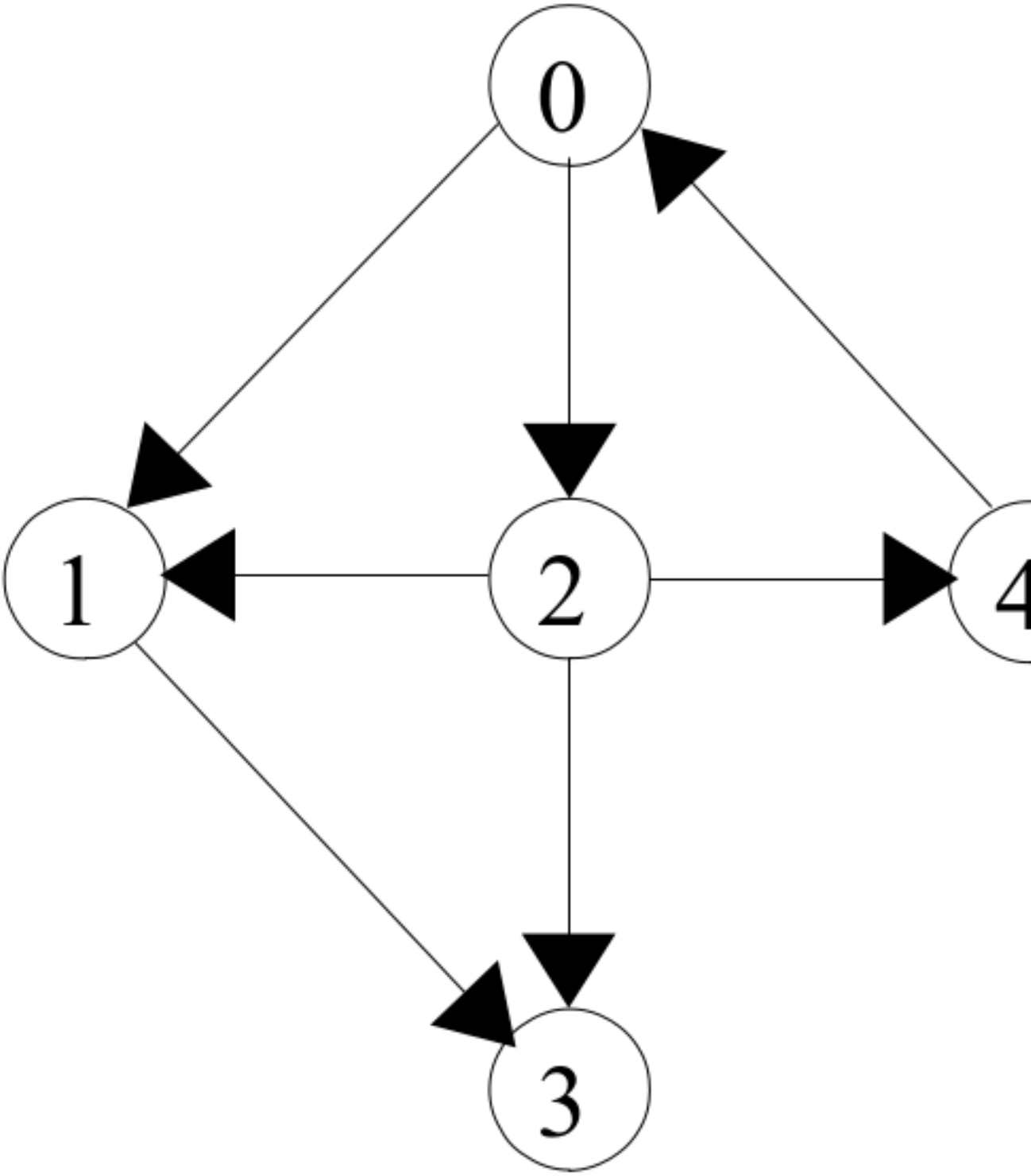
Exemple de sortida 9

```
T T T T F F F F F F
```

Informació del problema

Autoria: Jordi Esteve

Generació: 2026-01-25T19:57:42.190Z


0
1
2
4
3