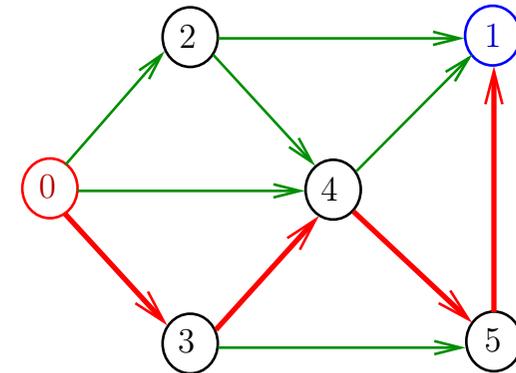


Problema: dados um digrafo G e dois vértices s e t decidir se existe um caminho de s a t

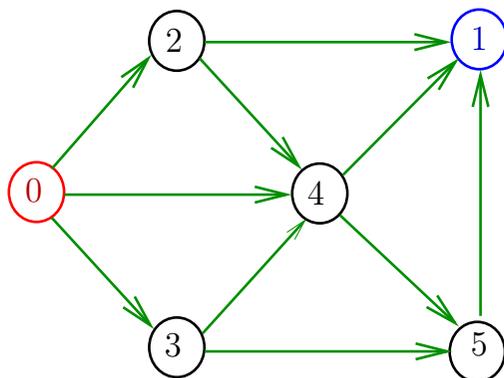
Exemplo: para $s = 0$ e $t = 1$ a resposta é SIM



Procurando um caminho

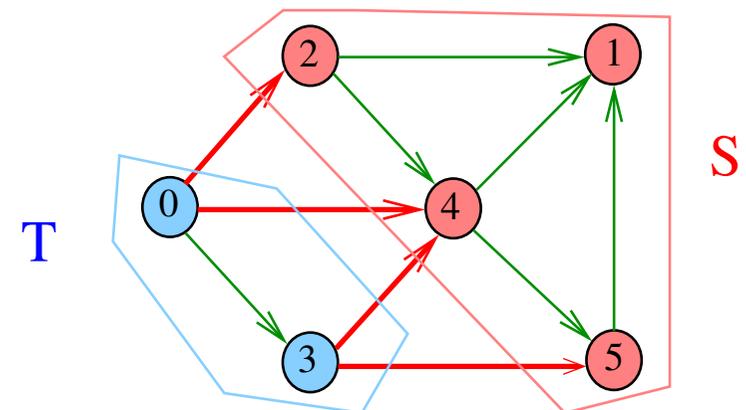
Problema: dados um digrafo G e dois vértices s e t decidir se existe um caminho de s a t

Exemplo: para $s = 5$ e $t = 4$ a resposta é NÃO



Certificado de inexistência

Exemplo: certificado de que não há caminho de 2 a 3



Conclusão

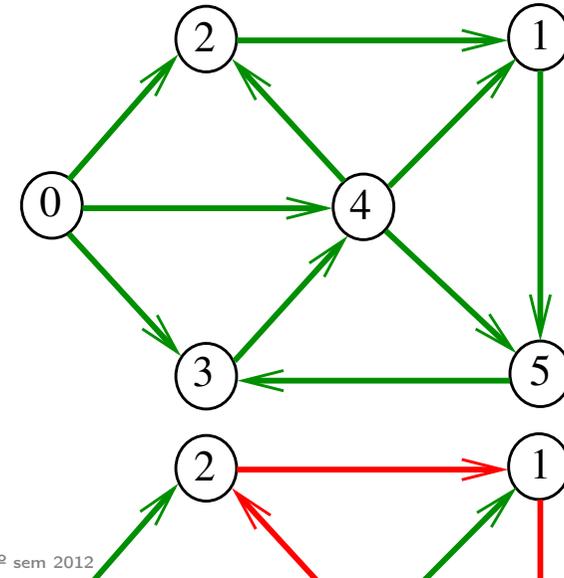
Para quaisquer vértices s e t de um digrafo, vale uma e apenas uma das seguintes afirmações:

- existe um caminho de s a t
- existe st -corte (S, T) em que todo arco no corte tem ponta inicial em T e ponta final em S .

Procurando um ciclo

Problema: decidir se dado digrafo G possui um ciclo

Exemplo: para o grafo a seguir a resposta é SIM



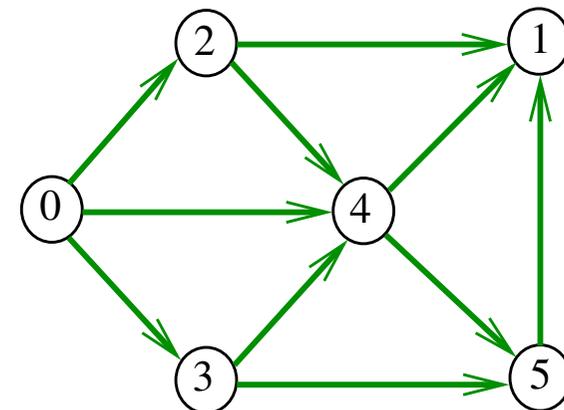
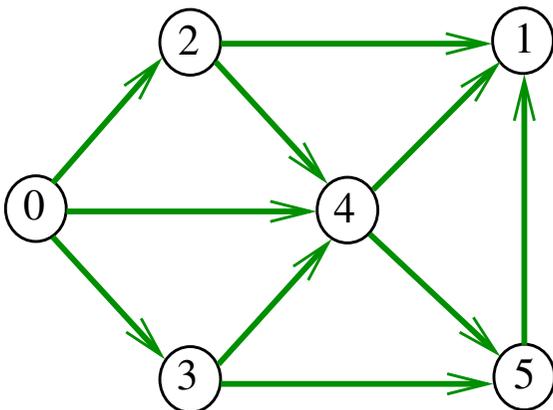
Ordenação topológica

i	0	1	2	3	4	5
$ts[i]$	0	3	2	4	5	1

Procurando um ciclo

Problema: decidir se dado digrafo G possui um ciclo

Exemplo: para o grafo a seguir a resposta é NÃO

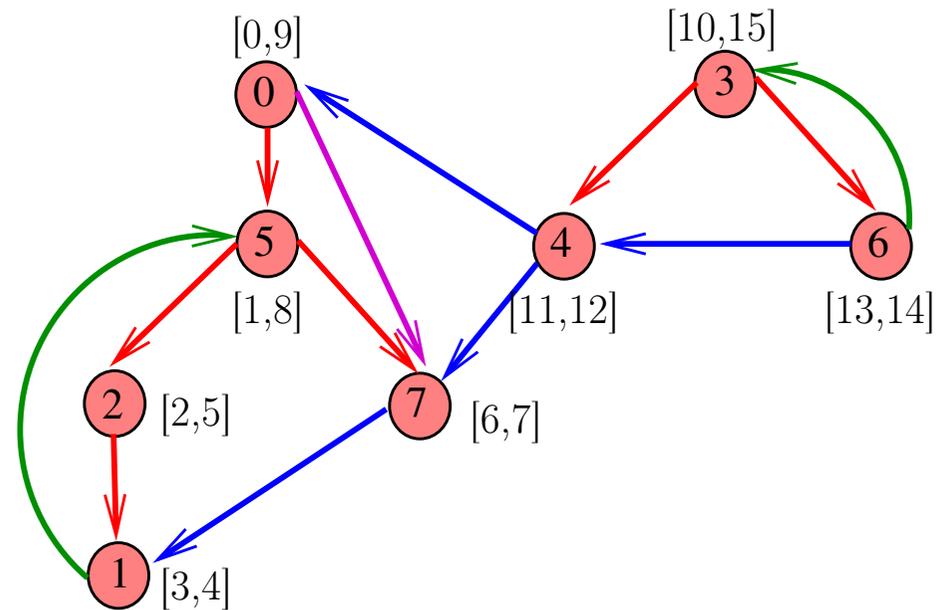


Conclusão

Para todo digrafo G , vale uma e apenas uma das seguintes afirmações:

- G possui um **ciclo**
- G é um DAG e, portanto, admite uma **ordenação topológica**

Floresta DFS

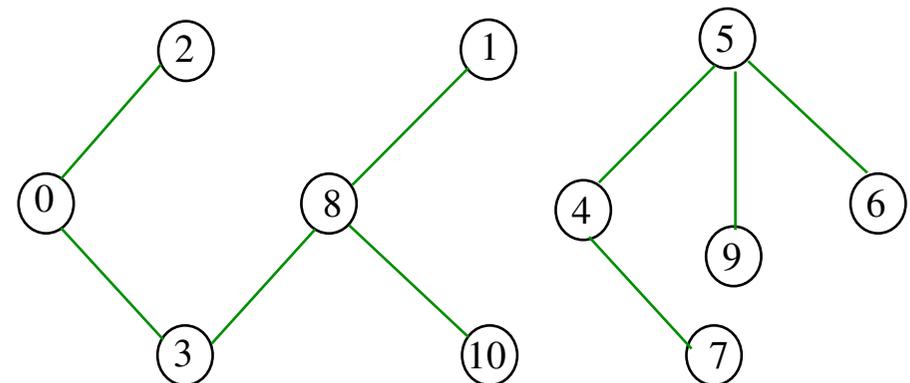


Florestas e árvores

Florestas

Uma **floresta** (= *forest*) é um grafo sem ciclos não-triviais

Exemplo:



Propriedades

Para cada par s, t de vértices de uma árvore existe um e um só caminho simples de s a t .

Toda árvore com V vértices tem exatamente $V-1$ arestas.

Conclusão

Para todo grafo G , vale uma e apenas uma das seguintes afirmações:

- G possui um **ciclo não trivial**
- G é uma floresta

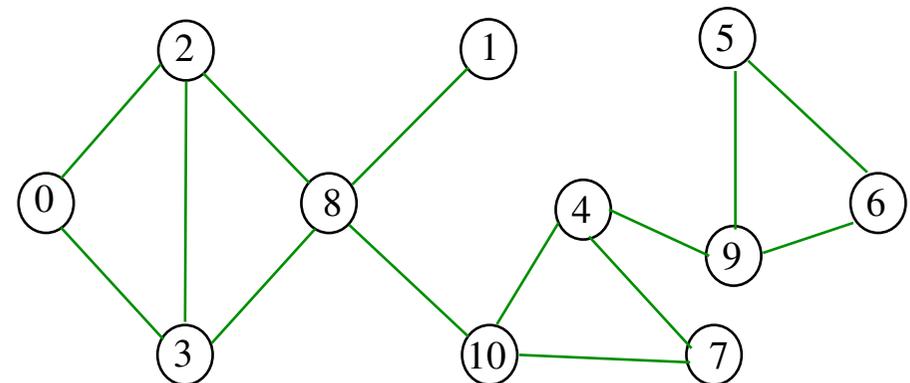
Componentes de grafos

S 18.5

Grafos conexos

Um grafo é **conexo** se e somente se, para cada par (s, t) de seus vértices, existe um caminho com origem s e término t .

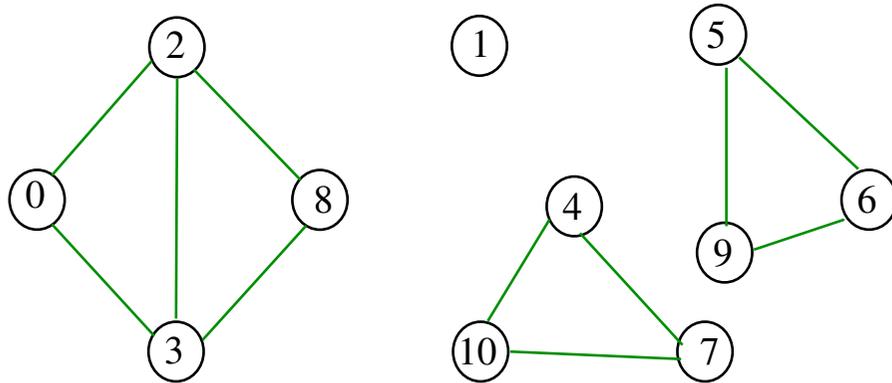
Exemplo: um grafo conexo



Componentes de grafos

Uma **componente** (= *component*) de um grafo é o subgrafo conexo maximal

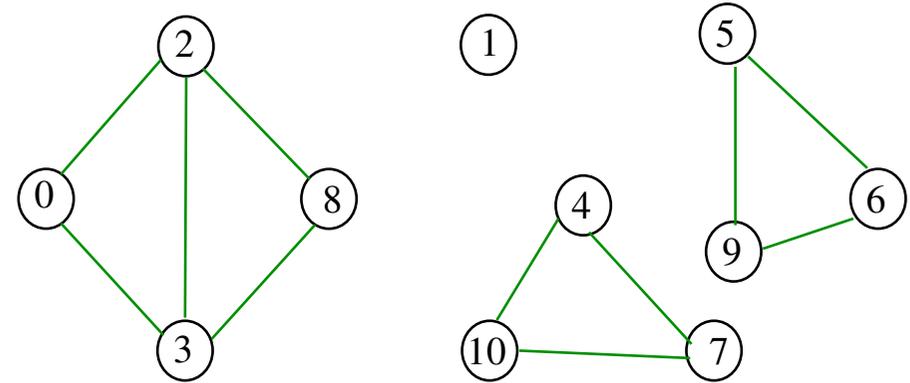
Exemplo: grafo com 4 componentes (conexos)



Contando componentes

Problema: calcular o número de componente

Exemplo: grafo com 4 componentes



Cálculo das componentes de grafos

A função abaixo devolve o número de componentes do grafo G .

```
#define maxV 10000
```

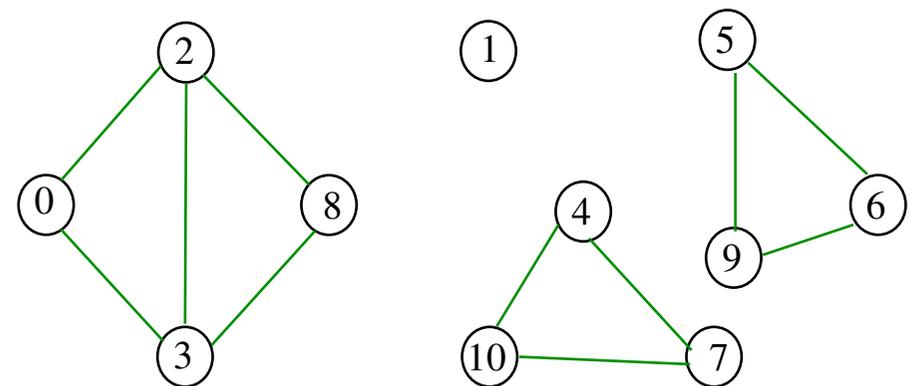
```
static int cc[maxV];
```

Além disso, ela armazena no vetor cc o número do componente a que o vértice pertence: se o vértice v pertence ao k -ésimo componente então $cc[v] == k-1$

```
int GRAPHcc (Graph G)
```

Exemplo

v	0	1	2	3	4	5	6	7	8	9	10
$cc[v]$	0	1	0	0	2	3	3	2	0	3	2



GRAPHcc

```
int GRAPHcc (Graph G) {  
    Vertex v; int id = 0;  
1   for (v = 0; v < G->V; v++) cc[v] = -1;  
2   for (v = 0; v < G->V; v++)  
3       if (cc[v] == -1)  
4           dfsRcc(G, v, id++);  
5   return id;  
}
```

dfsRcc

```
void dfsRcc (Graph G, Vertex v, int id){  
    link p;  
1   cc[v] = id;  
2   for (p=G->adj[v]; p!=NULL;p=p->next)  
3       if (cc[p->w] == -1)  
4           dfsRcc(G, p->w, id);  
}
```

Consumo de tempo

O consumo de tempo da função `GRAPHcc` é $O(V + A)$.

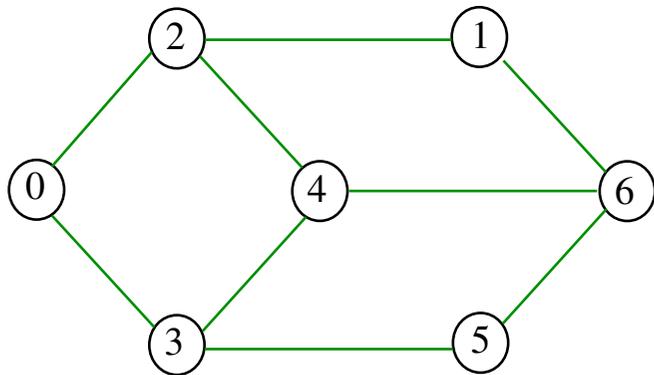
Grafos bipartidos e ciclos ímpares

S 18.5

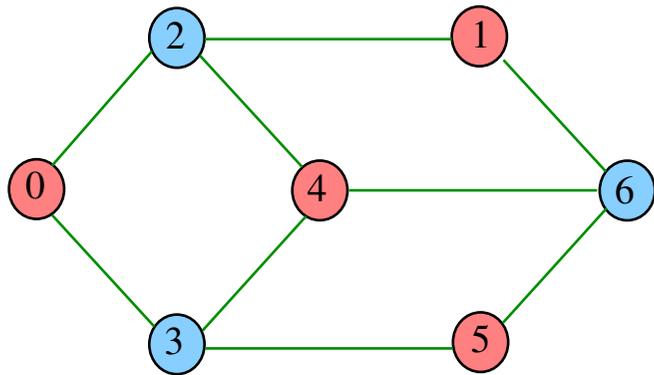
Bipartição

Um grafo é **bipartido** (= *bipartite*) se existe uma bipartição do seu conjunto de vértices tal que cada aresta tem uma ponta em uma das partes da bipartição e a outra ponta na outra parte

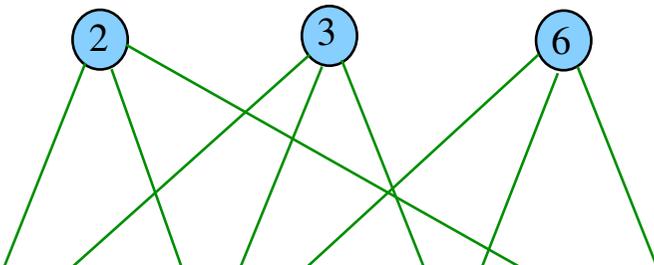
Exemplo:



Exemplo:



Exemplo:



Florestas e bipartições

Teorema 1.

Toda floresta é um grafo bipartido.

Prova: Basta mostrar que toda árvore é.

Escolha um vértice v , e, para todo vértice w , faça:

$$\text{cor}[w] = \text{dist}(v, w) \bmod 2.$$

Se u e w são adjacentes, o caminho de v ao mais distante passa pelo mais próximo e usa a aresta. Assim, as distâncias a v diferem de 1, e eles têm cores diferentes.