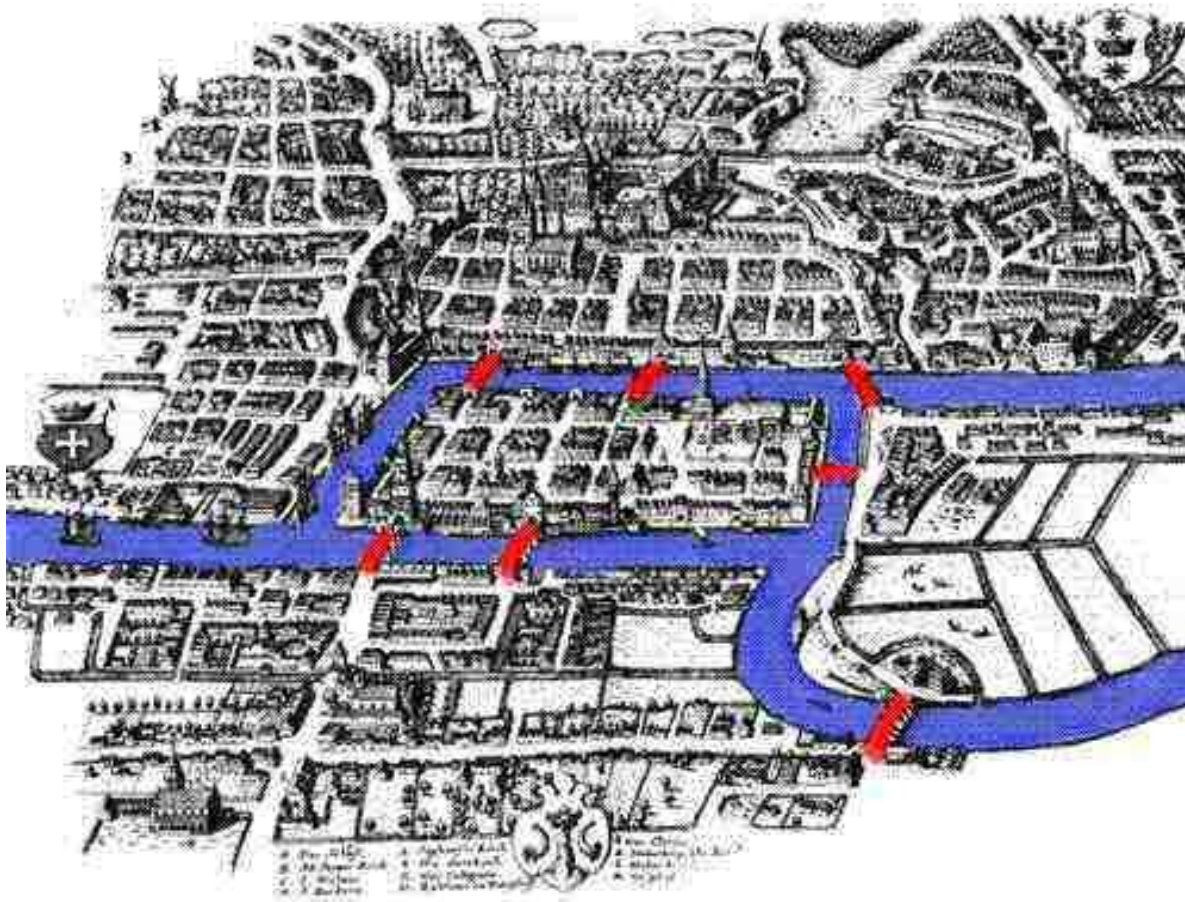


# Grafos – Conceitos e Algoritmos

Introdução, Definições  
Métodos de busca

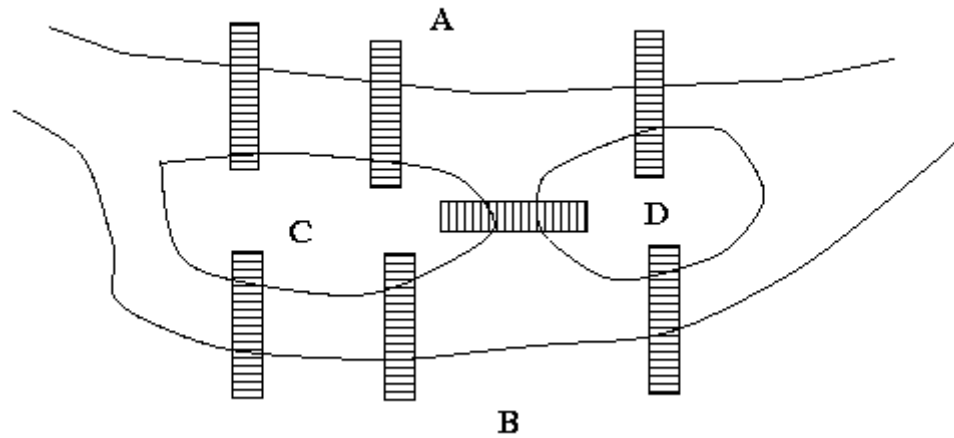
# Introdução

- Pontes de Königsberg: Existe um caminho que passe por todas as 7 pontes, um única vez sobre cada ponte ?

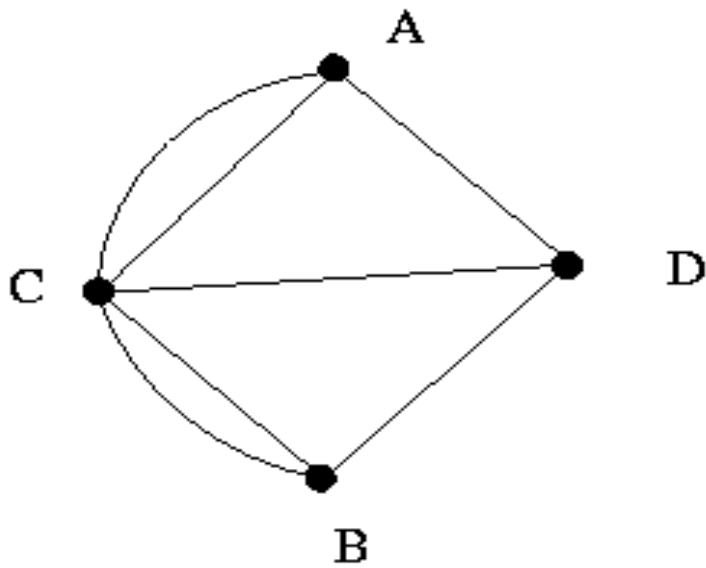


# Introdução

- Pontes de Königsberg



# Introdução



- Pontes de Königsberg
- Resposta: Não
- Como e porque? Leonhard Euler transformou os caminhos em retas e suas intersecções em pontos.
- Então percebeu que só seria possível atravessar o caminho inteiro passando uma única vez em cada ponte se houvesse exatamente zero ou dois pontos de onde saísse um número ímpar de caminhos.

# Definições

- Grafo – especifica um formato ou meta-modelo genérico para se modelar aspectos de um problema, de uma estrutura ou de um fenômeno.
- Grafos são importantes e úteis!
  - A grande maioria dos meta-modelos utilizados em computação é um grafo.
  - Muitos conjuntos de elementos físicos ou virtuais, seja em computação ou em outras áreas, possuem um mapeamento direto em uma estrutura de grafo.
  - Os estados e transições do processo de resolução de muitos problemas pode ser modelado como um grafo.

# Definições

- Meta-modelos ou ferramentas de modelagem que são grafos:
  - Exemplo:
    - Máquina de estados finitos
    - Diagrama entidade-relacionamento
    - Diagrama de classe UML
    - Diagrama atividades da UML
    - Processo de negócio em BPMN
    - Diagrama de componentes
    - Rede de Petri

# Definições

- Conjuntos de elementos que podem ser mapeados como grafos:
  - Mapa de cidades
  - Rede local de computadores
  - Conjunto de objetos que compõe um programa
  - Círculo social
  - Cruzamentos entre ruas de uma cidade
  - Rotina diária de uma pessoa
  - Programa orientado a objetos
  - Programa em linguagem imperativa.
  - WWW
  - Internet
  - Corporação

# Introdução

- Estados e transições de um processo de resolução de problema
  - Problema dos cântaros (8, 5, 3)
  - Problema da travessia (lobo, feno, cordeiro)



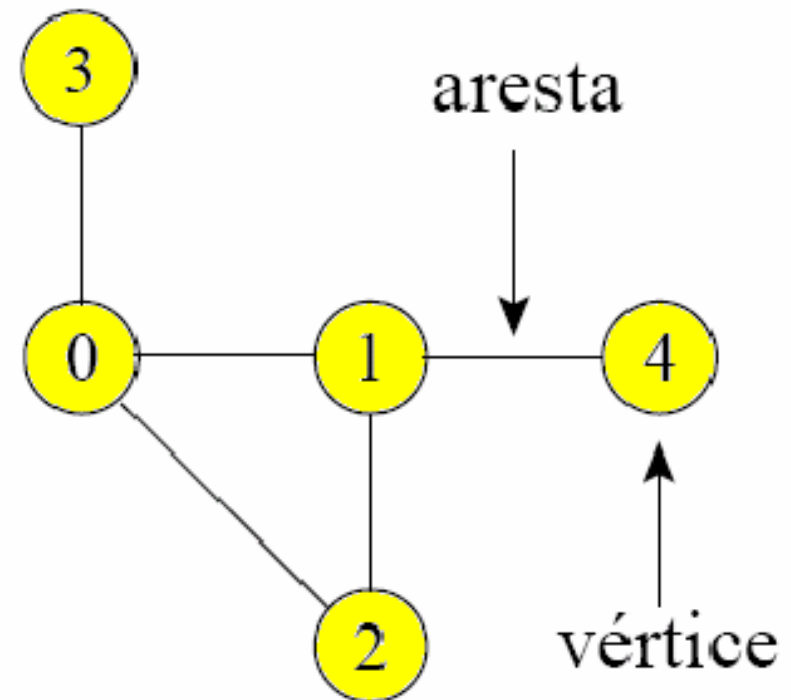
# Definições

- Um grafo é formado por conjuntos de dois objetos especiais:

- Vértices
- Arestas

Dois conjuntos finitos:

- Vértices  $V(G)$
- Arestas  $E(G)$
- Em geral, um grafo  $G$  é rep  
 $G = (V, E)$



# Definições

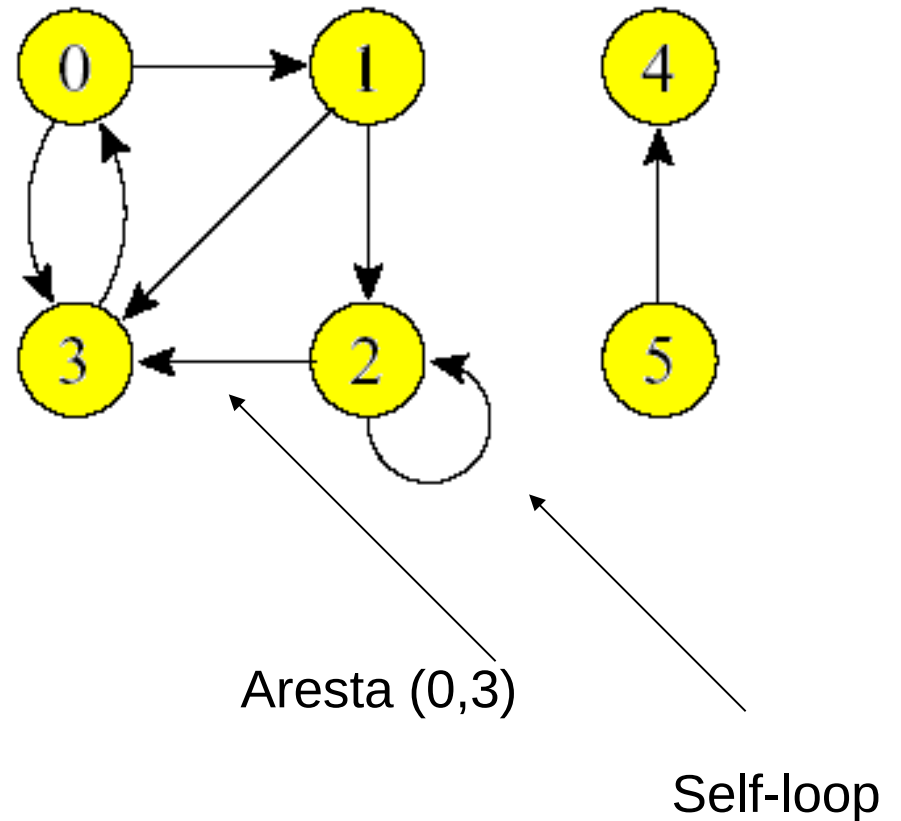
- Cada aresta está associada a um conjunto de um ou dois vértices, chamados nós terminais.
- Extremidade de uma aresta: vértice da aresta.
- Função aresta–extremidade: associa aresta a vértices.
- Laço (Loop): aresta somente com um nó terminal.
- Arestas paralelas: arestas associadas ao mesmo conjunto de vértices.
- Uma aresta é dita conectar seus nós terminais.
- Dois vértices que são conectados por uma aresta são chamados de adjacentes.
- Um vértice que é nó terminal de um laço é dito ser adjacente a si próprio.
- Uma aresta é dita ser incidente a cada um de seus nós terminais.
- Duas arestas incidentes ao mesmo vértice são chamadas de adjacentes.
- Um vértice que não possui nenhuma aresta incidente é chamado de isolado.
- Um grafo com nenhum vértice é chamado de vazio.

# Definições

- Exemplo: Jogo dos palitos para 2 pessoas começa com um conjunto de  $N$  palitos ( $N$  inteiro maior que 0). Cada jogador, na sua vez, pode pegar 1, 2 ou 3 palitos; quem retira o último perde.
  - Modelar este problema como um grafo
  - Mostrar que se  $N$  for igual 15, 7 ou 10, o primeiro jogador pode ter uma estratégia em que sempre vence.

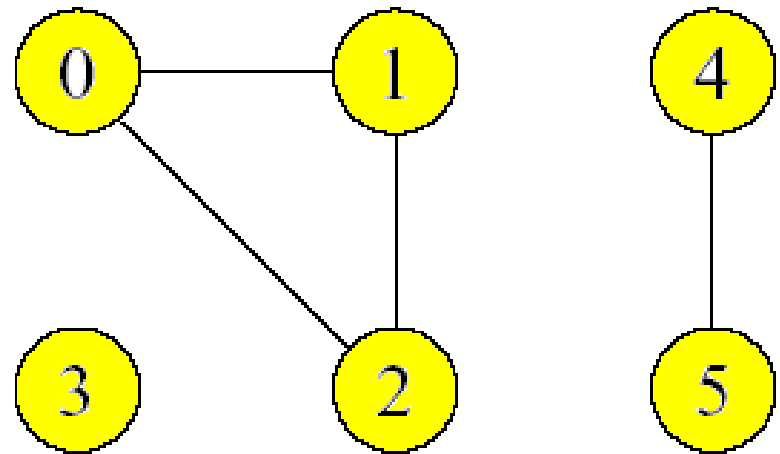
# Definições

- Grafo Direcionado:
  - Digrado ou direcionado
  - As Arestas possuem sentido
  - Uma aresta parte de um vértice e chega em outro
  - Podem existir self-loops



# Definições

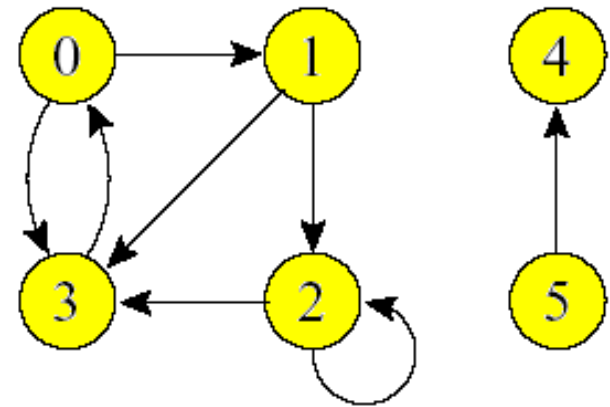
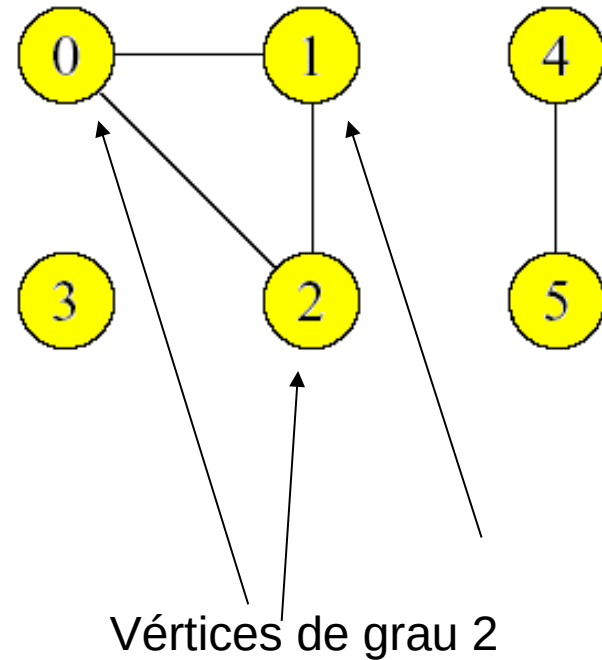
- Grafos Não Direcionados
  - As arestas  $(u,v)$  e  $(v,u)$  são consideradas uma única aresta
  - Self-loops não existem



Exemplo de grafo  
não direcionado

# Definições

- Grau de um vértice
  - Em grafos não direcionados é o número de arestas que incide nele
  - Um laço é contado duas vezes.
  - Vértices de grau zero são ditos isolados ou não conectados
  - Em grafos direcionados pode se calcular o grau de entrada (in-degree) e o grau de saída (out-degree) de um vértice. O grau é a soma do in-degree com o out-degree



Vértice 2 –  
in-dregree= 2  
out-ddegree=2

# Definições

- Grau total de um grafo: Soma dos graus de todas vertices.
- Teorema: Grau total do grafo  $G = \text{grau}(G) = 2$  vezes o número de arestas de  $G$ .
  - Logo, O grau total de um grafo é um número par.
- Teorema: Em qualquer grafo  $G$ , existe um número par de vértices de grau ímpar.

# Definições

- É possível ter um grafo com quatro vértices de graus 1, 1, 2, e 3?  
Não. O grau total deste grafo é 7, que é um número ímpar.
- Definição: Um grafo simples é um grafo que não possui laços nem arestas paralelas. Num grafo simples, uma aresta com vértices (nós terminais)  $u$  e  $v$  é representada por  $uv$ .



# Definições

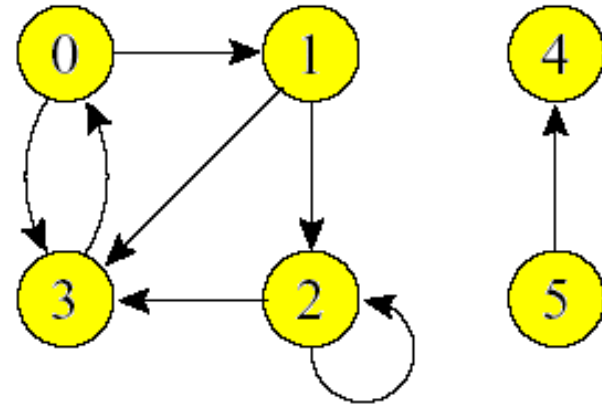
- É possível ter um grafo simples com quatro vértices de graus 1, 1, 3, e 3?

Não.

- Prova (por contradição):
  - Suponha que exista um grafo simples  $G$  com quatro vértices de graus 1, 1, 3, e 3. Chame  $a$  e  $b$  os vértices de grau 1, e  $c$  e  $d$  os vértices de grau 3.
  - Como  $\text{grau}(c) = 3$  e  $G$  não possui laços ou arestas paralelas, devem existir arestas que conectam  $c$  aos vértices  $a$ ,  $b$  e  $d$ .
  - Pelo mesmo raciocínio devem existir arestas que conectam  $d$  aos vértices  $a$ ,  $b$  e  $c$ .
  - Mas o  $\text{grau}(a) \geq 2$  e  $\text{grau}(b) \geq 2$ , o que contradiz a suposição que estes vértices têm grau 1..
  - Daí. . a suposição inicial é falsa e, conseqüentemente, não existe um grafo simples com quatro vértices com graus 1, 1, 3, e 3.

# Definições

- Ciclo
  - Um caminho  $(v_0, v_1, \dots, v_k)$  forma um ciclo se  $v_0 = v_k$  e  $k \geq 1$
  - Ciclo simples:  $(v_1, \dots, v_k)$  são vértices distintos

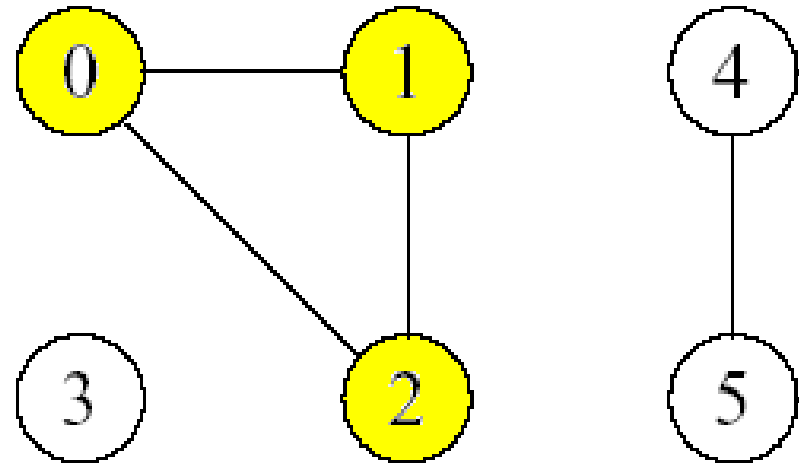


Exemplos de Ciclos Simples:

$\{0, 3, 0\}$ ,  $\{0, 1, 3, 0\}$ ,  $\{2, 2\}$

# Definições

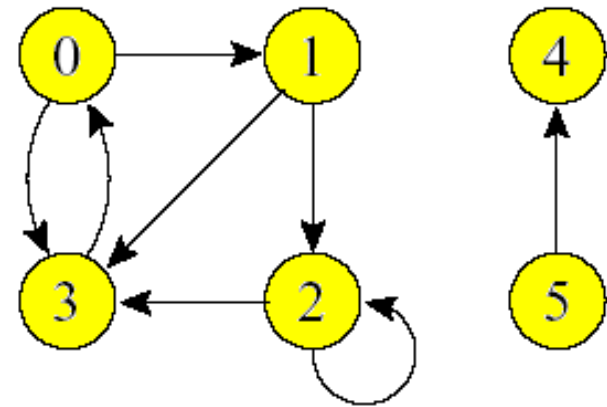
- Em um grafo não direcionado:
  - Um caminho  $(v_0, v_1, \dots, v_k)$  forma um ciclo se  $v_0 = v_k$  e o caminho possui pelo menos 3 arestas
  - O ciclo é simples se os seus vértices são distintos



O caminho  $\{0,1,2\}$  é um ciclo

# Definições

- Caminho entre dois vértices  $x$  e  $y$ :
  - Seqüência de vértices  $(v_0, v_1 \dots v_k)$  tal que
    - $v_i$  pertence a  $A$  para  $i=0 \dots k$
    - $x=v_0$  e  $y=v_k$
  - O comprimento do caminho é dados por  $k$  que indica o número de arestas do caminho

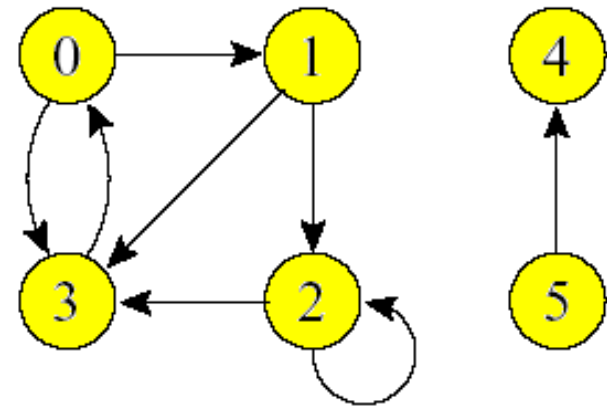


Caminho entre 0 e 3:  
 $\{0, 1, 2, 3\}$ ,  $v_k=3$ ,  $k=3$   
Comprimento  $=k=3$

A aresta 3 é alcançável a partir  
Da aresta 0 via aresta 2

# Definições

- Caminho entre dois vértices  $x$  e  $y$ :
  - Seqüência de vértices  $(v_0, v_1 \dots v_k)$  tal que
    - $v_i$  pertence a  $A$  para  $i=0 \dots k$
    - $x=v_0$  e  $y=v_k$
  - O comprimento do caminho é dados por  $k$  que indica o número de arestas do caminho

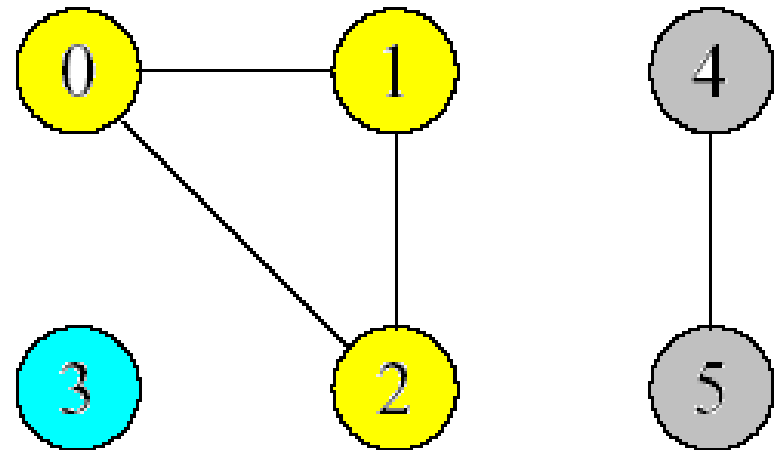


Caminho entre 0 e 3:  
 $\{0, 1, 2, 3\}$ ,  $v_k=3$ ,  $k=3$   
Comprimento  $=k=3$

A aresta 3 é alcançável a partir  
Da aresta 0 via aresta 2

# Definições

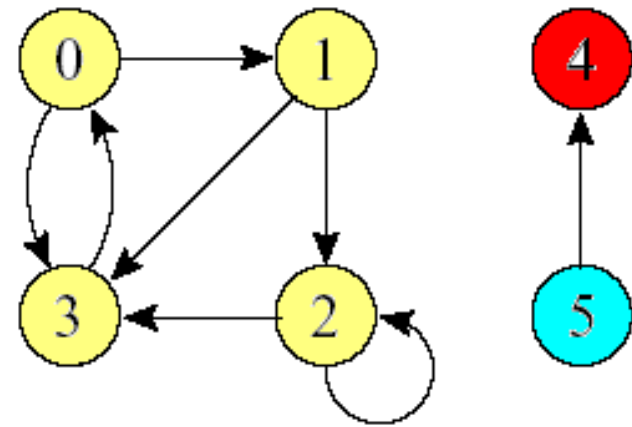
- Componentes Conectados
  - Um grafo não direcionado é conectado se existe um caminho entre todos os possíveis pares de vértices
  - Componentes conectados são porções conectadas de um grafo
  - Logo, um grafo não direcionado é conectado se ele possui apenas um componente conectado



Componentes:  
 $\{3\}$  ,  $\{0,1,2\}$ ,  $\{4,5\}$

# Definições

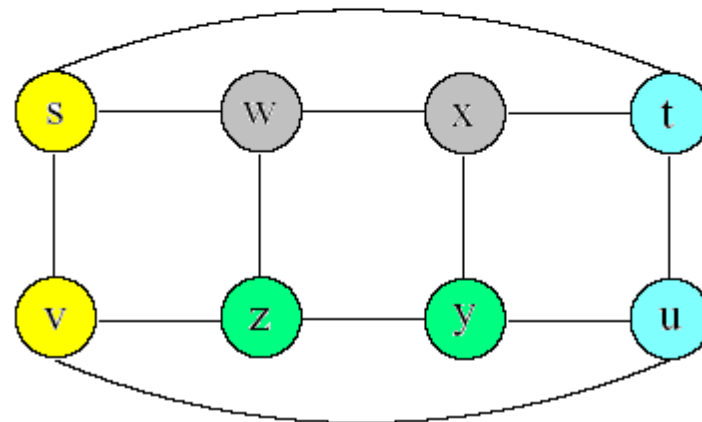
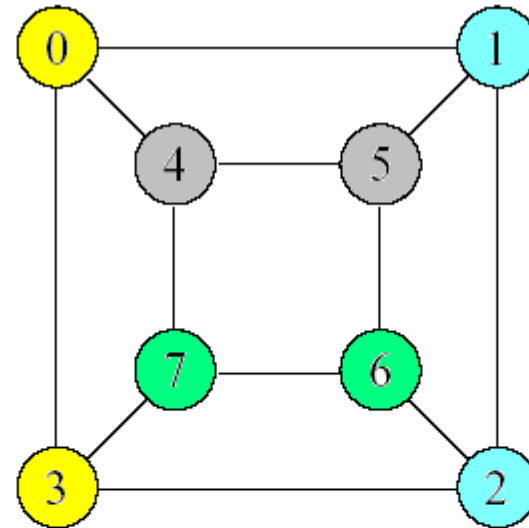
- Componentes Fortemente conectados
  - Um grafo direcionado  $G$  é fortemente conectado se cada dois vértices quaisquer de  $G$  são alcançáveis a partir um do outro
  - Um componente fortemente conectado de um grafo direcionado é um conjunto de vértices mutuamente alcançáveis



Componentes fortemente conectados:  
 $\{0,1,2,3\}$   $\{4\}$ ,  $\{5\}$

# Definições

- Grafos Isomorfos:
  - $G = (V, A)$  e  $G' = (V', A')$  são isomorfos que existe uma função bijetora  $f : V \rightarrow V'$  tal que  $(u, v) \in A$  se e somente se  $(f(u), f(v)) \in A'$



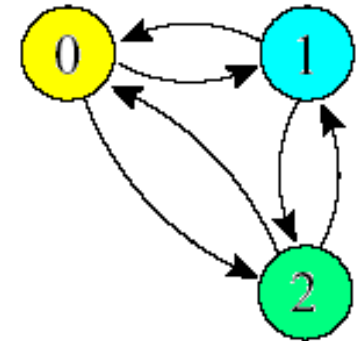
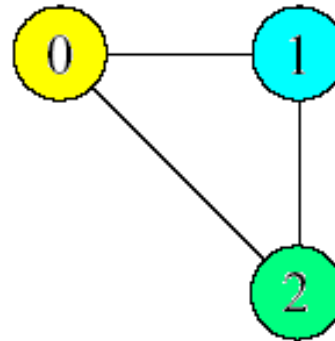


# Definições

- SubGrafo:
  - Um grafo  $G'=(V',A')$  é um subgrafo de  $G=(V,A)$  se  $V'$  está contido em  $V$  e  $A'$  está contido em  $A$

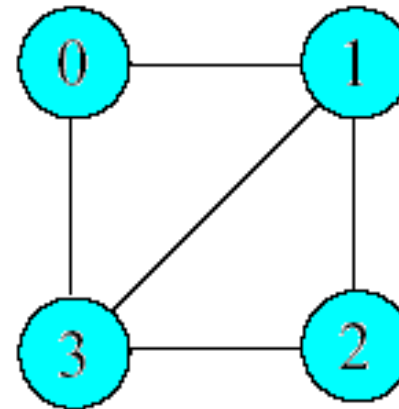
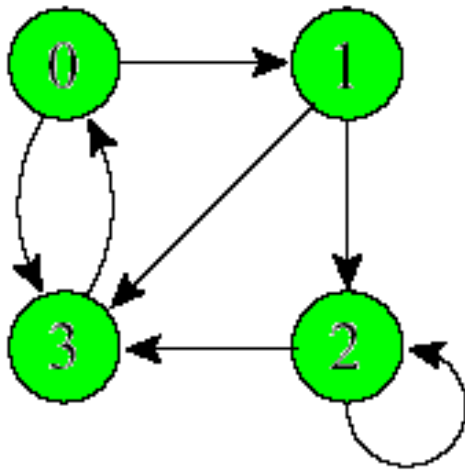
# Definições

- Versão direcionada de um grafo não direcionado:
  - Cada aresta não direcionada é substituída por duas arestas direcionadas



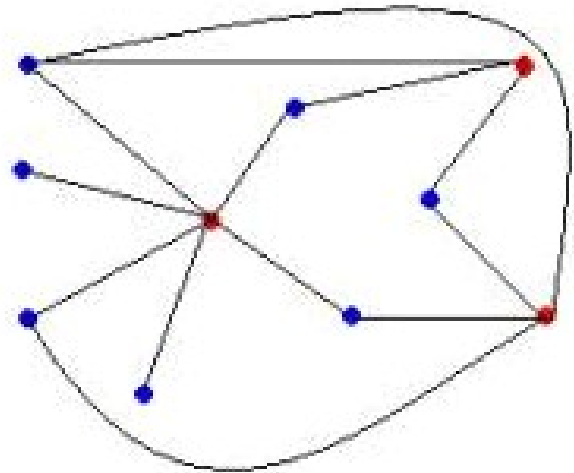
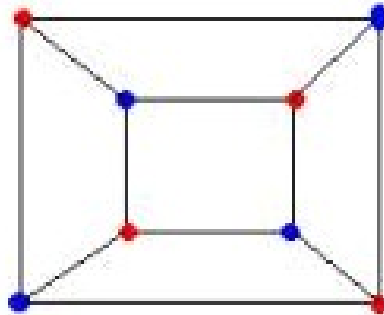
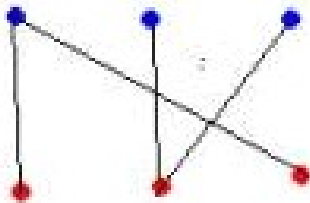
# Definições

- Versão não direcionada de um grafo direcionado
  - A versão não direcionada é formada eliminando a direção das arestas e os self-loops



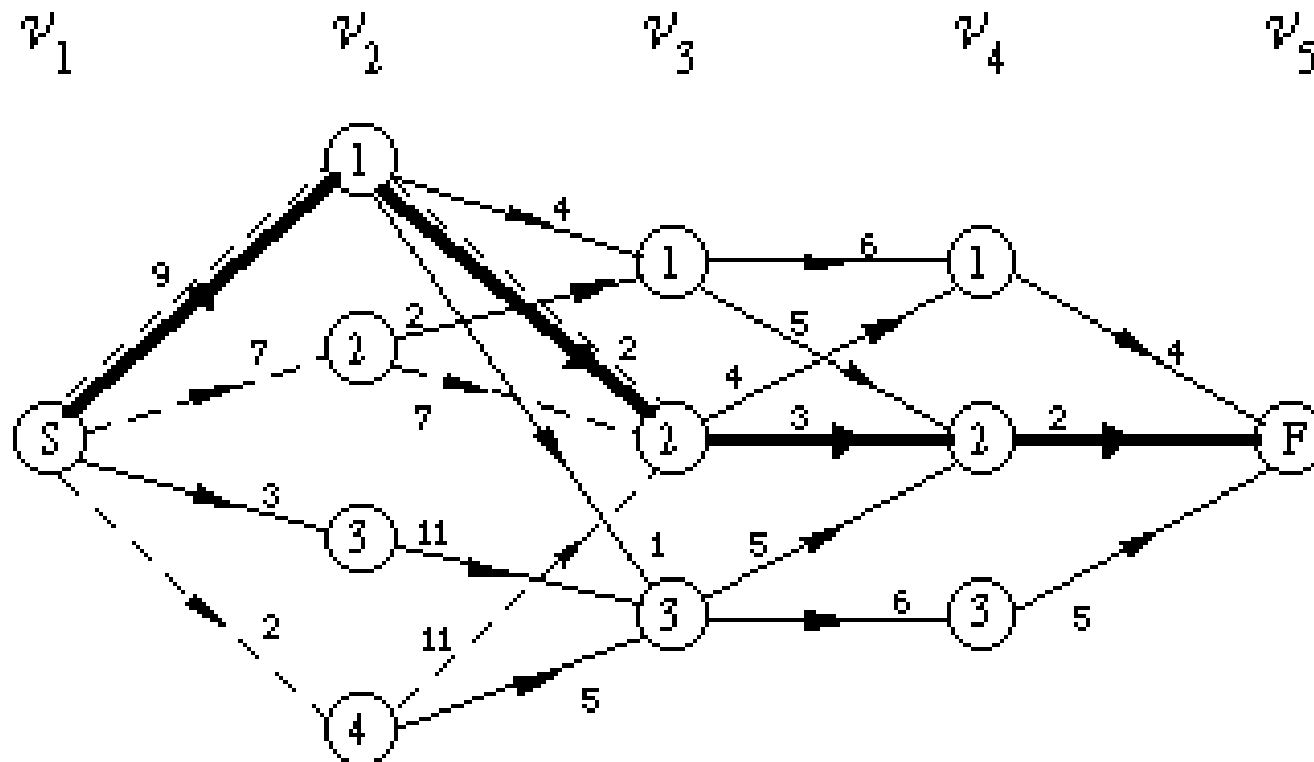
# Definições

- Classificações de grafos:
  - Grafo ponderado: Possui pesos associados às arestas
  - Grafo bipartido: É um grafo  $G=(V,A)$  que pode ser particionado em dois grafos  $G_1=(V_1,A)$  e  $G_2=(V_2,A)$ , onde todas as arestas ligam os dois conjuntos de vértices  $V_1$  e  $V_2$ .



Grafos bipartidos

# Definições



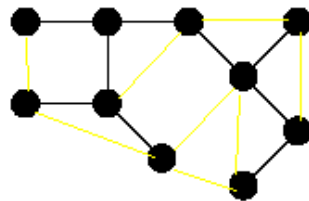
Grafo Ponderado

# Definições

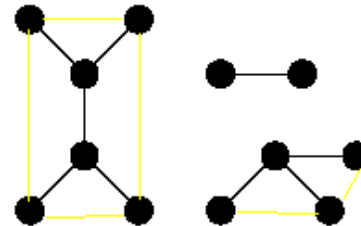
- Grafo completo:
  - Todas os seus pares de vértices são adjacentes
  - Um grafo completo não direcionado possui
$$\frac{(|V|^2 - |V|)}{2} = |V|(|V| - 1)/2$$
 arestas
  - O número de grafos diferentes que podem ser obtidos a partir de  $V$  vértices é dados por
$$2^{|V|(|V| - 1)/2}$$

# Definições

- Árvore : Grafo não direcionado acíclico e conectado;
- Floresta: Grafo não direcionado acíclico conectado ou não;
- Árvore Geradora: Uma árvore geradora de um grafo  $G$  é um Subgrafo de  $G$  que contem todos os vértices de  $G$  e forma uma árvore;
- Floresta Geradora: Subgrafo de  $G$  que contem todos os vértices de  $G$  e forma uma floresta.



(a)



(b)

# Caminhando em grafos: Busca em Profundidade

- Busca em Profundidade (Depth-first search): Caminha no grafo visitando todos os seus vértices
- Estratégia: Procura ir sempre o mais profundo no grafo



# Busca em Profundidade

- As arestas são exploradas a partir do vértice  $v$  mais recentemente descoberto que ainda tem arestas não descobertas saindo dele;
- Quando todas as arestas de  $v$  tiverem sido exploradas a busca anda para trás para explorar arestas que saem do vértice a partir do qual  $v$  foi descoberto

# Busca em Profundidade

- Todos os vértices são inicializados com branco
- Quando um vértice é visitado pela primeira vez ele torna-se cinza
- Quando sua lista de adjacentes foi totalmente explorada ele torna-se preto

# Busca em Profundidade

- Tempo de descoberta:  $d[v]$ 
  - É o momento em que o vértice  $v$  foi visitado pela primeira vez
- Tempo de término do exame da lista de adjacentes:  $t[v]$ 
  - É o momento em que a visita a toda lista de vértices adjacentes a  $v$  foi concluída
- $d[v]$  e  $t[v]$  são inteiros entre 1 e  $2V$ , onde  $V$  é o número de vértices do vetor

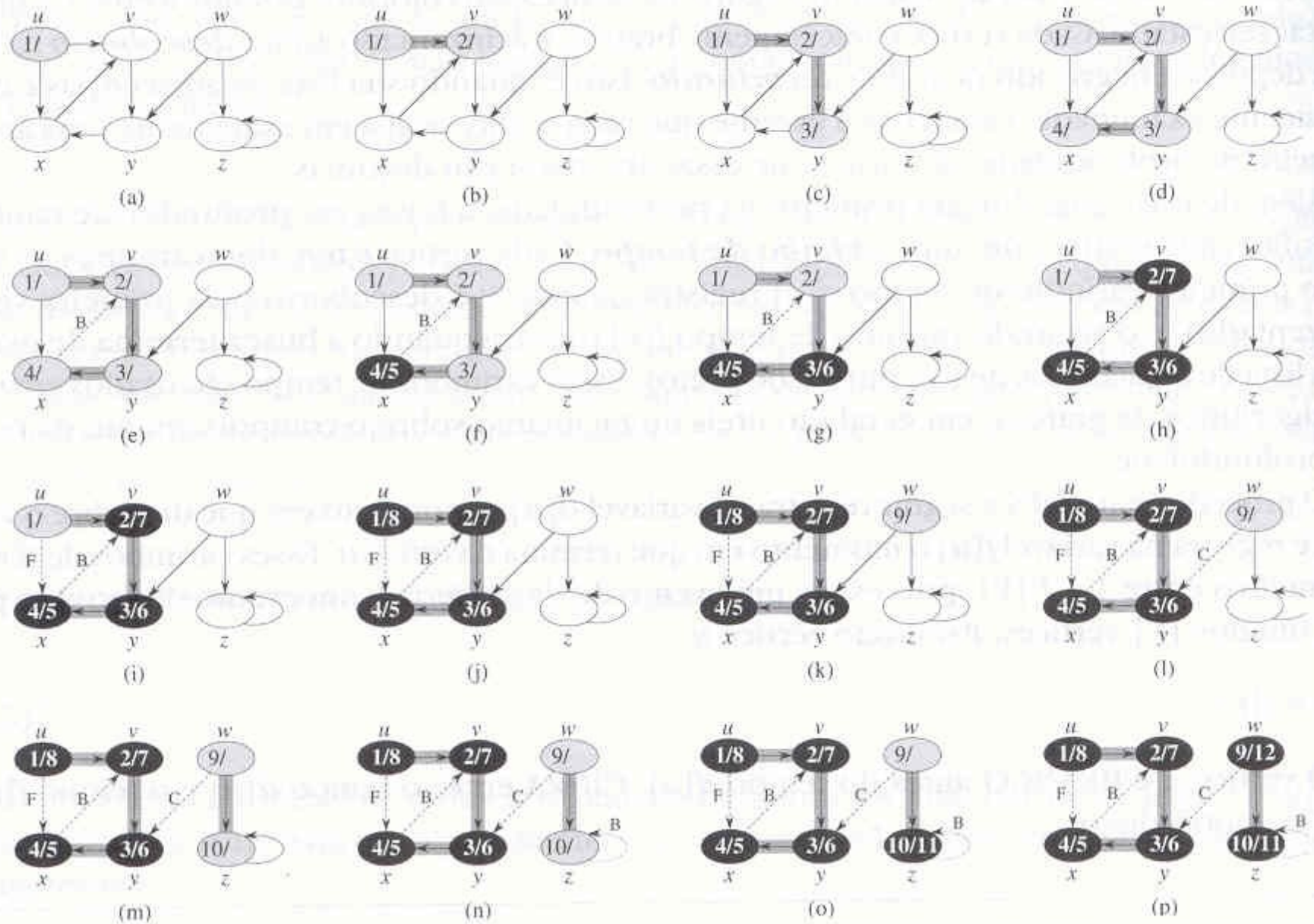
# Busca em Profundidade: Algoritmo

```
Algoritmo DFS(G: grafo)
  para cada vértice u de G faça
    cor[u] ← branco
    pred[u] ← -1
  fim para
  para cada vertice u de G faça
    se (cor[u] = branco) então
      visita(u)
    fim se
  fim para
```

# Busca em Profundidade - Algoritmo

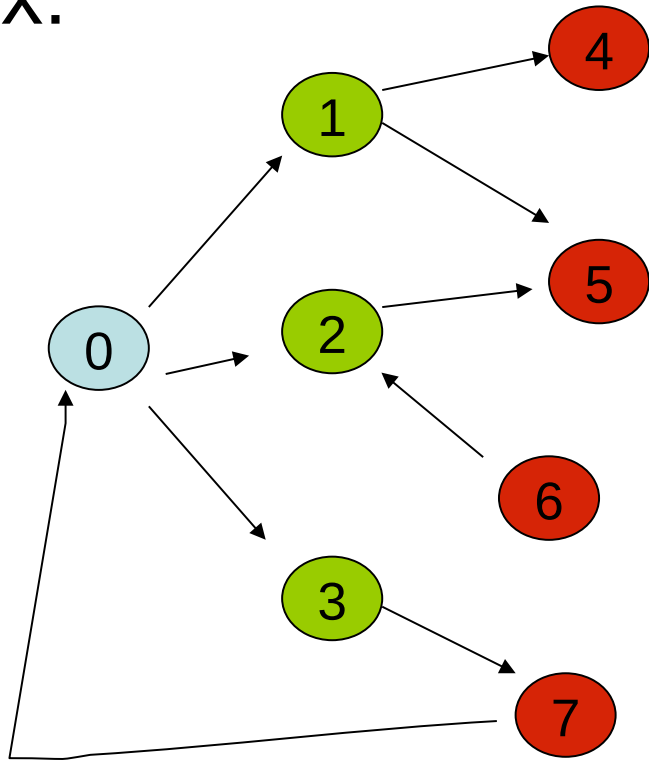
```
Visita(u: vertice)
  cor[u] ← cinza
  tempo ← tempo + 1
  d[u] ← tempo
  para cada v adjacente a u faça
    se cor[v] = branco entao
      pred[v] ← u
      visita(v)
  fim se
fim para
cor[u] ← preto
tempo ← tempo + 1
t[u] ← tempo
fim Visita
```

A Figura 22.4 ilustra o andamento de DFS sobre o grafo mostrado na Figura 22.2.



# Busca em Largura

- Expande-se o conjunto de vértices descobertos uniformemente pelas adjacências do nó recém descoberto:
- Ex:



Descobre o vértice 0  
Descobre os adjacentes de 0  
Descobre os adjacentes de 1  
Descobre os adjacentes de 2  
Descobre os adjacentes de 3  
Descobre os adjacentes de 5  
Descobre os adjacentes de 6  
Descobre os adjacentes de 7

# Busca em Largura

- Na busca em largura o algoritmo descobre todos os vértices a uma distância  $k$  do vértice de origem antes de descobrir os que estão a uma distância  $k+1$
- O grafo pode ser direcionado ou não direcionado



# Busca em Largura

- Algoritmo
  - Cada vértice é colorido de branco, cinza ou preto.
  - Todos os vértices são inicializados com branco.
  - Quando um vértice é descoberto pela primeira vez ele torna-se cinza.
  - Vértices cujos adjacentes são todos descobertos tornam-se pretos.
  - Se  $(u,v) \in A$  e o vértice  $u$  é preto, então  $v$  tem que ser cinza ou preto.
  - Vértices cinzas podem ter adjacentes brancos.

# Busca em Largura

VisitaBFS(G)

para cada vértice u de G faça

cor[u] ← BRANCO

d[u] ← infinito

antecessor[u] ← nil

fim para

para cada vertice u de G faça

se cor[u] ← BRANCO então

BFS(G,u)

fim para

Vértice	Antecessor
0	nil
1	0
2	0
3	0
4	1
5	1
6	2
7	3

BFS(G,s)

cor[s] ← cinza

d[s] ← 0

EsvaziaFila(Q)

Insere(Q,s)

Enquanto (FilaVazia(Q) == FALSO) faça

u ← remove(Q)

para cada v adjacente a u faça

se (cor[v] = BRANCO) então

cor[v] = CINZA

d[v] = d[u] + 1

antecessor[v] = u

insere(Q,v)

fim se

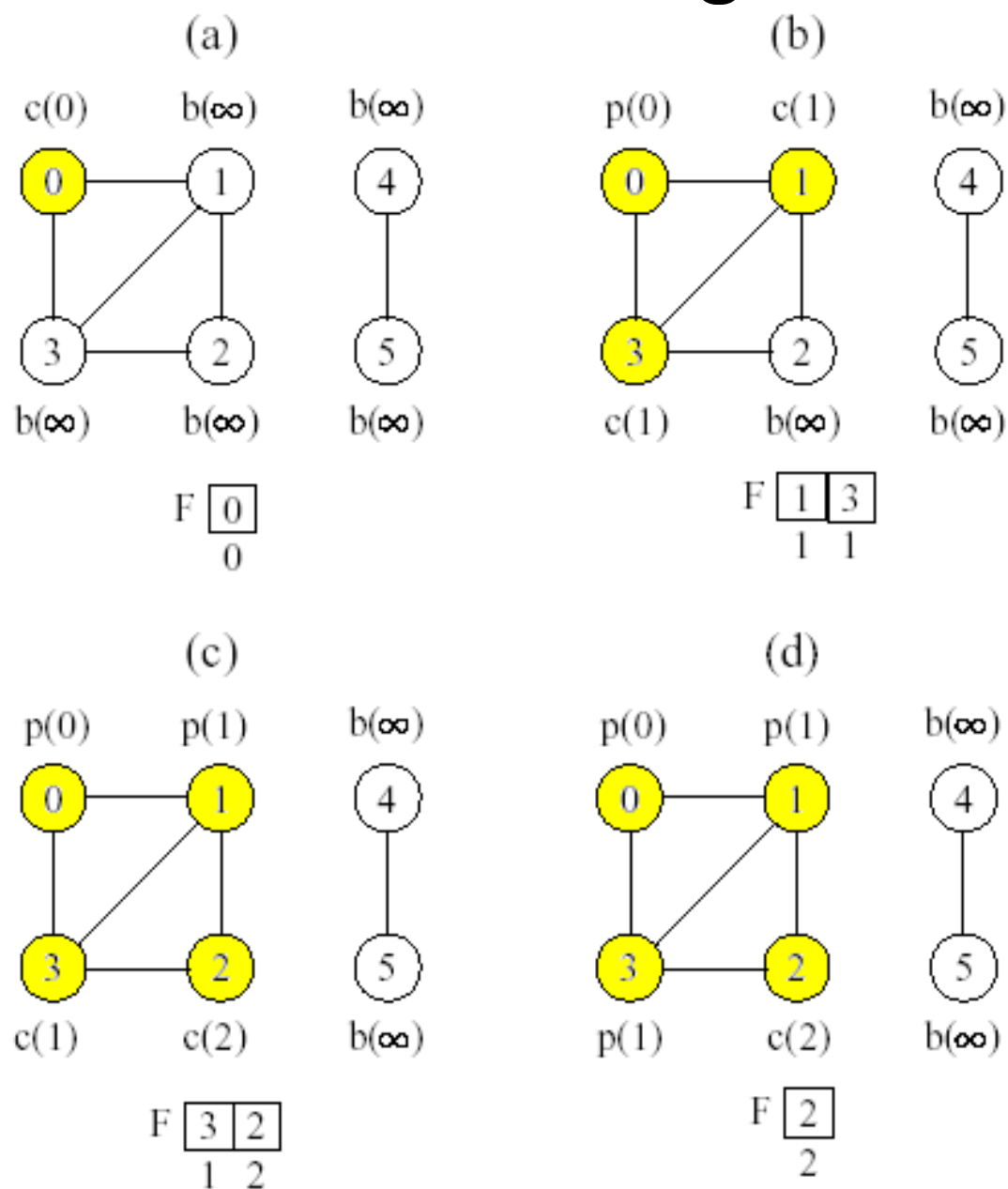
fim para

cor[u] ← PRETO

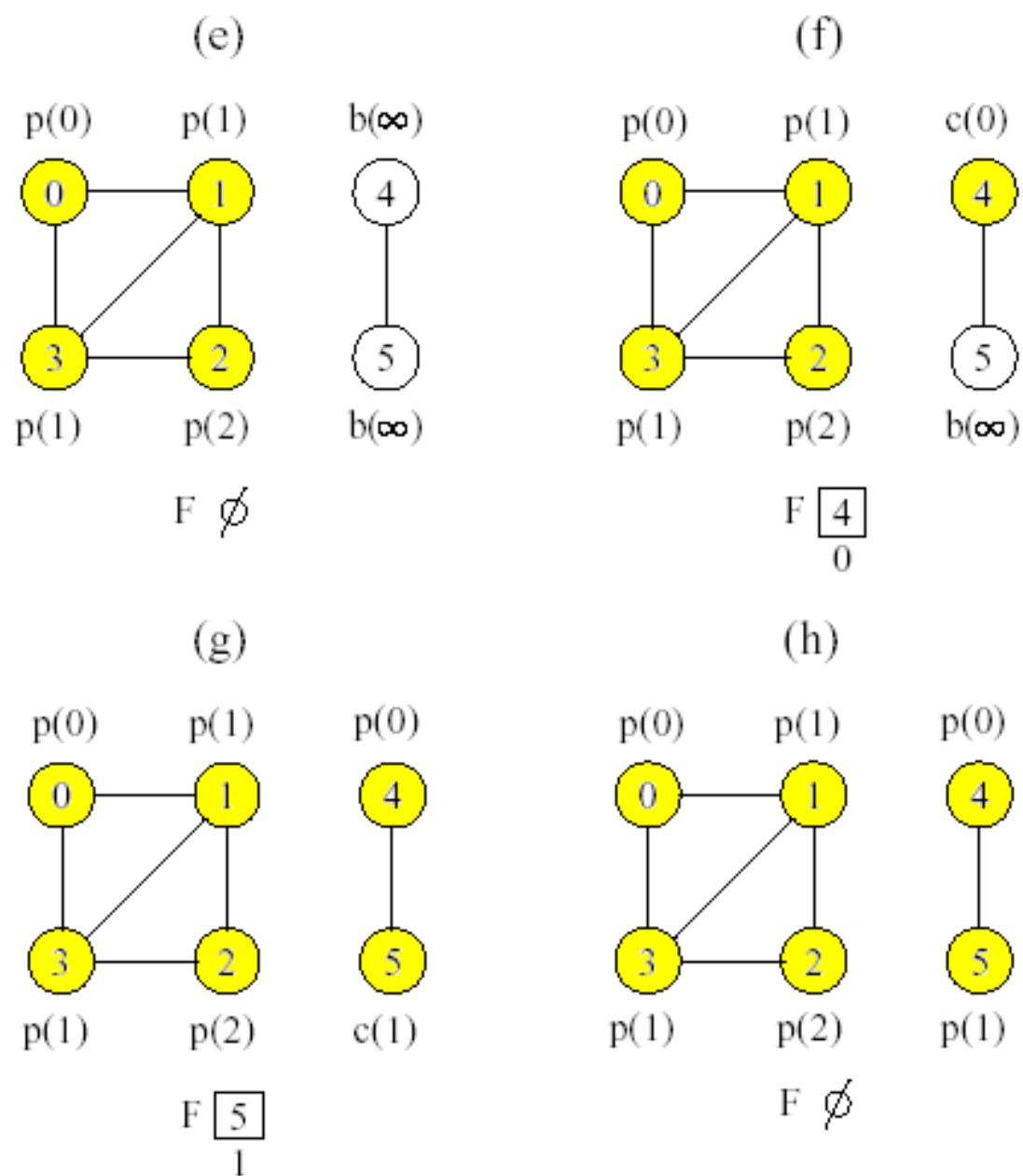
fim Enquanto

Fim BFS

# Busca em Largura



# Busca em Largura



# Caminhos mais curtos

- O BFS encontra o caminho mais curto entre dois vértice  $u$  e  $v$ .
- O caminho entre dois vertices quaisquer fica armazenado no vetor antecessor

# Caminhos mais curtos

- Ex: para imprimir o caminho mais curto entre um vértice  $v$  e o vértice de origem da busca:

```
ImprimeCaminho(Origem, v)
    se (Origem = v) entao
        imprima(origem)
    fim se
Senao
    se (antecessor[v] != nil)
        imprima("Não existe caminho de v até a origem")
    fim se
senao
    ImprimaCaminho(Origem, Antecessor[v])
    Imprima(v)
fim senao
```