Governo Federal

Ministério da Educação

Universidade Federal do Maranhão

A Universidade que Cresce com Inovação e Inclusão Social

Algoritmos em Grafos
Busca em Largura (BFS)

Estrutura de Dados II

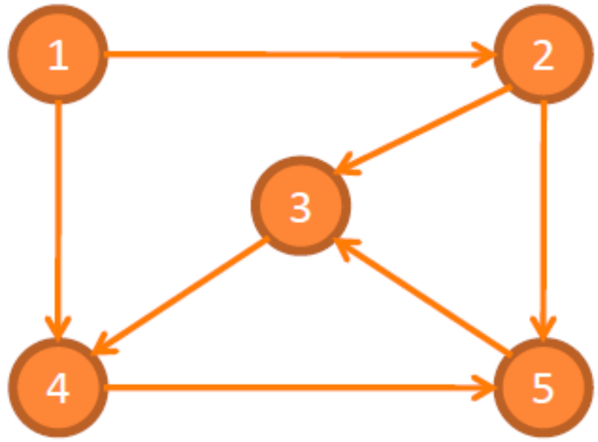
Slides baseados nas notas de aula do Prof. Geraldo Braz - DEINF/UFMA

Algoritmos Sobre Grafos

- **Busca em Largura**
  - Caminhos mais curto
- **Busca em Profundidade**
  - Classificação de arestas
  - Verificação de Grafo acíclico
- **Ordenação Topológica**
  - Componentes Fortemente Conectados
- **Árvore Geradora Mínima**
  - Prim e Kruskal
- **Algortimo de Dijkstra**

Percurso em Grafos

- **Um caminho ou percurso será fechado se a última ligação for adjacente ao vértice inicial**
- **A notação de um caminho é feita usando pares de vértices**
- **Um percurso no grafo abaixo: P(1,2,3,4,5) = ((1,2),(2,3),(3,4),(4,5),(5,3)**



Percurso em Grafos

- **Validação de cada vértice ou aresta**
- **Cópia de um grafo ou conversão de um tipo em outro**
- **Contagem do número de vértices ou arestas**
- **Determinação de componentes conexas**
- **Determinação de caminhos entre dois vértices, se existirem**
- **Eficiência – um vértice não pode ser visitado repetidamente**
- **O percurso deve ser feito de modo que não se perca nada**

Percurso em Grafos

- **Os vértices devem ser marcados quando visitados pela primeira vez**
- **Cada vértice apresenta três estados**
  - Não visitado
  - Visitado
  - Completamente explorado
- **Mantém-se uma estrutura de dados com todos os vértices já visitados mas não completamente explorados (Fila ou Pilha)**
- **Arestas não orientadas podem ser consideradas duas vezes**

Busca em Largura (BFS)

- **Expande a fronteira entre vértices descobertos e não descobertos uniformemente através da largura da fronteira.**

- **O algoritmo descobre todos os vértices a uma distância k do vértice origem antes de descobrir qualquer vértice a uma distância k + 1.**

- **O grafo G(V,E) pode ser direcionado ou não direcionado.**
- **Breadth-first search (BFS)**

Busca em Largura (BFS)

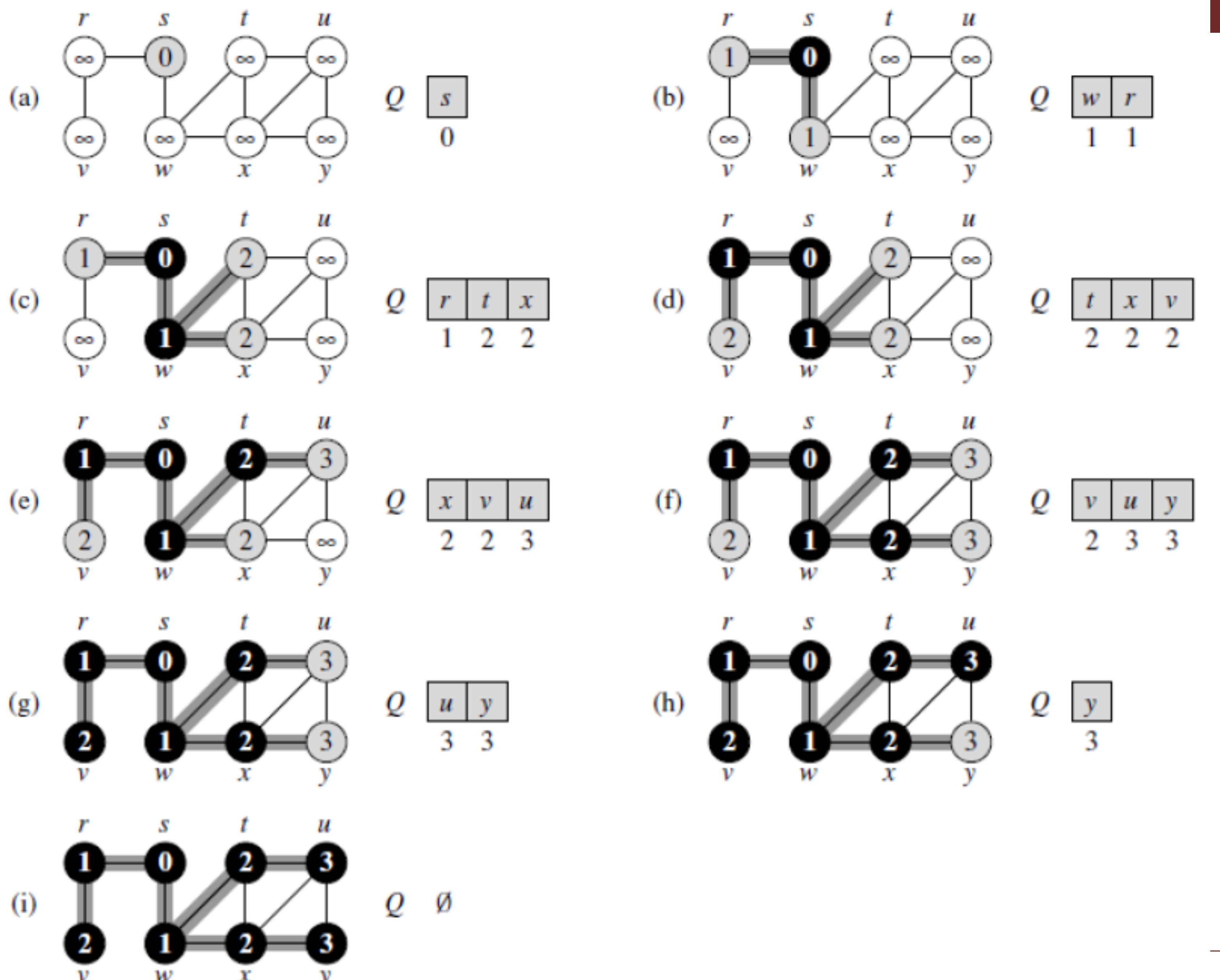
- **Cada vértice é colorido de branco, cinza ou preto.**
- **Todos os vértices são inicializados branco.**
  – Quando um vértice é descoberto pela primeira vez ele torna-se cinza.
  – Vértices cinza e preto já foram descobertos, mas são distinguidos para assegurar que a busca ocorra em largura.
  – Se (u, v) Є E e o vértice u é preto, então o vértice v tem que ser cinza ou preto.
  – Vértices cinza podem ter alguns vértices adjacentes brancos, e eles representam a fronteira entre vértices descobertos e não descobertos.

- **O algoritmo calcula a distância (menor número de arestas) para todos os vértices acessíveis a partir de *s*.**

Busca em Largura (BFS)

- **Passos:**
  - Escolhe um vértice para o início do caminhamento
  - Visita os vértices adjacentes marcando-os como visitados
  - Coloca cada um dos vértices em uma fila
  - Após visitados os vértices adjacentes, o primeiro da fila se torna o próximo vértice inicial
  - Termina quando todos os vértices tenham sido visitados ou o vértice procurado seja encontrado

Busca em Largura (BFS)



Algoritmo BFS

```
BFS(G, s)
for cada vértice u ∈ V[G] – {s}
    do cor[u] ← BRANCO
       d[u] ← ∞
       α[u] ← NIL
cor[s] ← CINZA
d[s] ← 0
α[s] ← NIL
Q ← 0
ENQUEUE(Q, s)
while Q ≠ 0
    do u ← DEQUEUE(Q)
       for cada v ← Adj[u]
           do if cor[v] = BRANCO
                 then cor[v] ← CINZA
                      d[v] ← d[u] + 1
                      π[v] ← u
                      ENQUEUE(Q, v)
       cor[u] ← PRETO
```

Alfa (pai de todos os vertices) = antecessor

Pi = predecessor

Inicializa

Primeiro nó do grafo passado passa a ser cinza

Coloca na FIFO todos os cinzas

Enquanto existir elementos cinzas

Atende o primeiro da FIFO e procura pelos adjacentes

Se houverem adjacentes que não foram varridos (brancos)

Aumenta a distância do adjacente, levando em consideração o pai.

Coloca na FIFO

Ao Final, pinta de preto

BFS - Análise

- Custo de inicialização do primeiro anel no método BFS é $O(|V|)$.
- Custo do segundo anel é também $O(|V|)$.
- Enfileirar e desenfileirar têm custo $O(1)$, logo, o custo total com a fila é $O(|V|)$.
- Cada lista de adjacentes é percorrida no máximo uma vez, quando o vértice é desenfileirado.
- Desde que a soma de todas as listas de adjacentes é $O(|E|)$, o tempo total gasto com as listas de adjacentes é $O(|E|)$.
- Complexidade total: é $O(|V| + |E|)$.

Caminho mais curto

- **A busca em largura obtém o caminho mais curto de u até v.**
- **O procedimento *BFS* contrói uma árvore de busca em largura que é armazenada na variável antecessor (pi).**
- **O programa abaixo imprime os vértices do caminho mais curto entre o vértice origem e outro vértice qualquer do grafo, a partir do vetor antecessor obtido na busca em largura.**

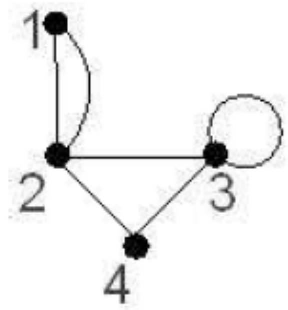
```
PRINT-PATH(G, s, v)
if v = s
   then imprimir s
   else if π[v] = NIL
           then imprimir "nenhum caminho de" s "para" v "existente"
           else PRINT-PATH(G, s, π[v])
                imprimir v
```

Aplicações

- **Web Crawling – Google**
  - Web = grafo, hiperdocumentos = nós e hiperlinks = arestas
- **Redes Sociais (facebook, Linkedin)**
  - Busca de amigos / busca amigos dos amigos
- **Broadcast de Rede**
- **Garbage colletor - Java**

Exercícios

Seja o grafo G a seguir.



[Poscomp 2013]

Com base nesse grafo, considere as afirmativas a seguir.

I. O grafo G é conexo.

II. A matriz de adjacências do grafo G é dada por $\begin{bmatrix} 0 & 2 & 0 & 0 \\ 2 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$

III. O grau do vértice 2 é igual a 2.

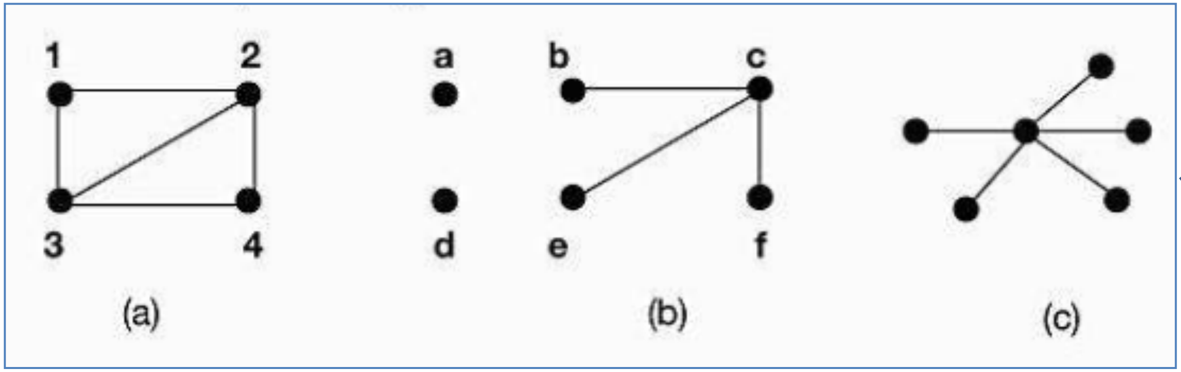
IV. O grafo G é denotado como Grafo Simples.

Assinale a alternativa correta:

a) Somente as afirmativas I e II são corretas.
b) Somente as afirmativas I e IV são corretas.
c) Somente as afirmativas III e IV são corretas.
d) Somente as afirmativas I, II e III são corretas.
e) Somente as afirmativas II, III e IV são corretas.

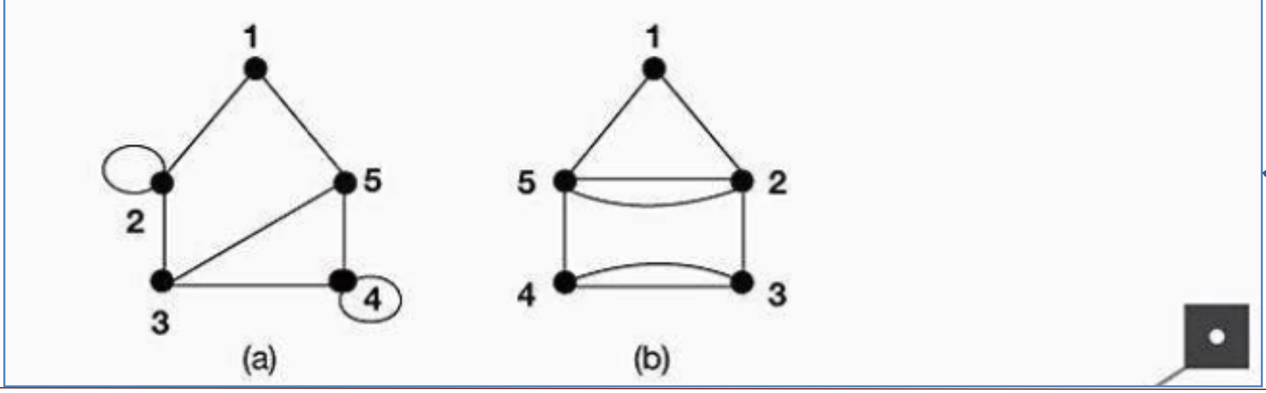
Exercício 1 - comentário

- **Grafo Simples:**
  - é um grafo que não possui laços ou arestas múltiplas



SIMPLES

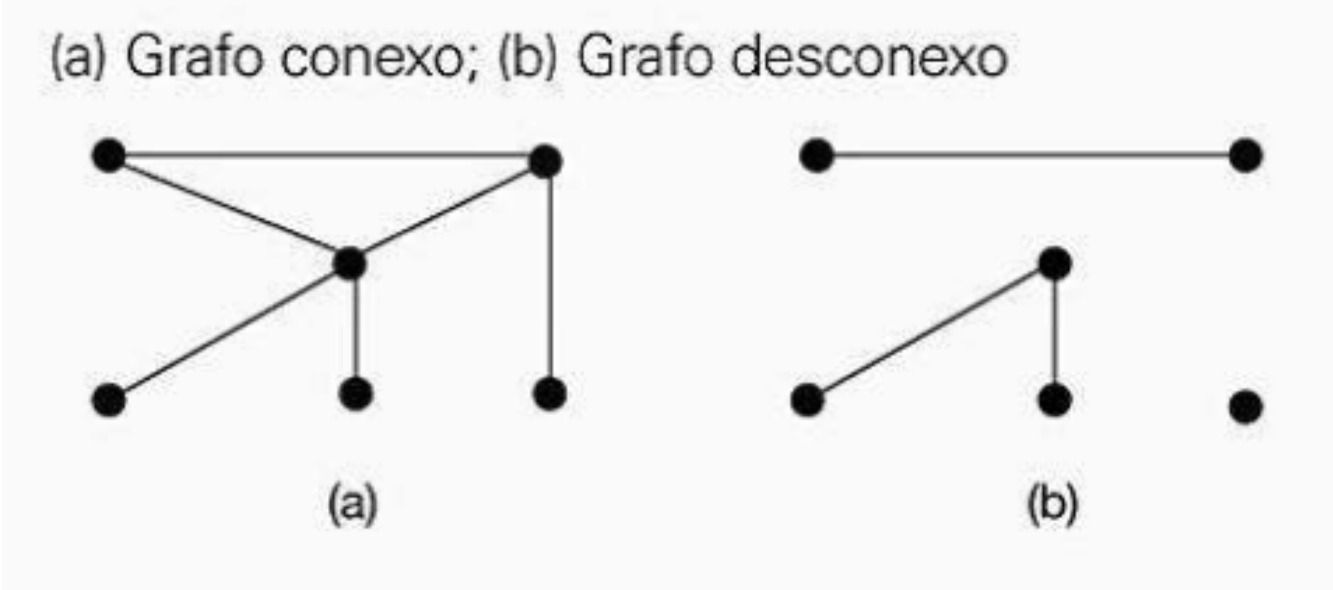
(a) Grafo com laços; (b) Grafo com arestas múltiplas, ou seja, multigrafo



NÃO SIMPLES

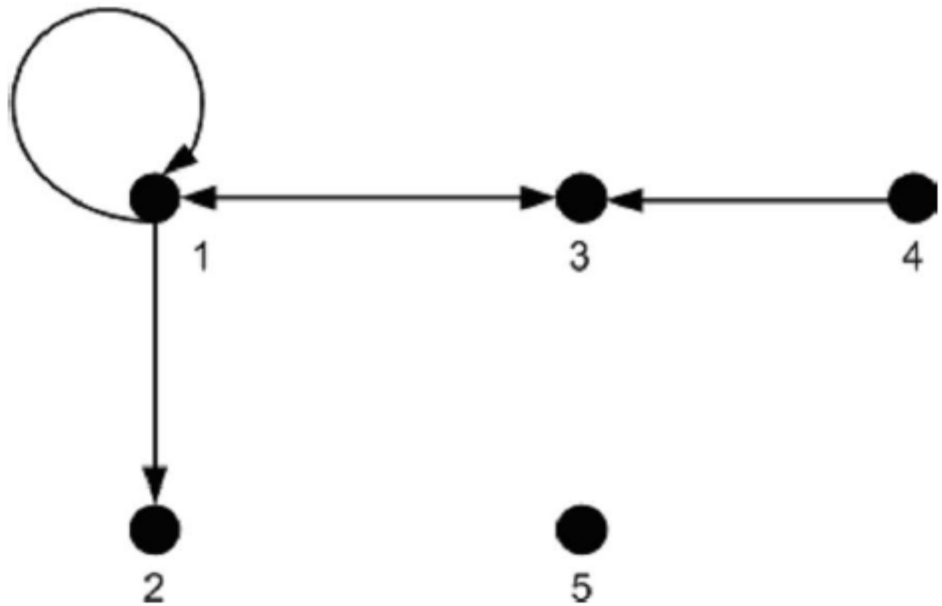
Exercício 1 - comentário

- **Grafo conexo e desconexo**
  - se existe um caminho para cada par de vértices de G. Caso contrário é chamado de desconexo.



Exercicio 2 [poscomp 2011]

Considere o grafo a seguir.



O grafo representa a relação:

a) R = {(1, 1), (1, 2), (1, 3), (3, 1), (4, 3)}
b) R = {(1, 1), (1, 2), (1, 3), (3, 1), (3, 4)}
c) R = {(1, 1), (1, 3), (2, 1), (3, 1), (3, 4)}
d) R = {(1, 1), (1, 2), (1, 3), (3, 4), (4, 3)}
e) R = {(1, 1), (1, 3), (2, 1), (3, 1), (4, 3)}

Exercícios

3) Construa o grafo orientado para o percurso P(1,2,3,4,5,6)=((1,3),(3,2),(2,4),(4,6),(6,5),(5,1))

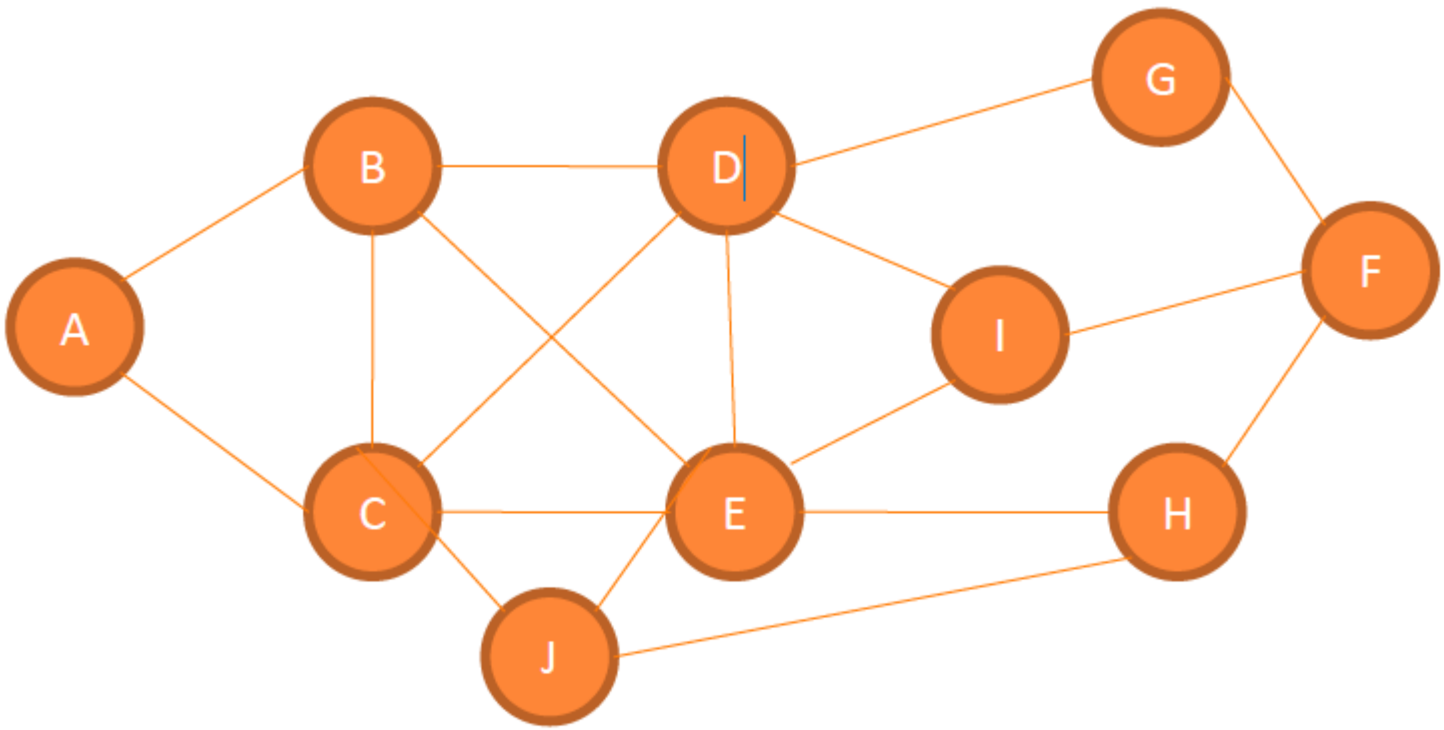
4) Implemente um programa em C/Java que receba um grafo na forma de matriz e o converta em lista de adjacências.

Exercício

5) Dado o grafo a seguir qual o maior comprimento que a fila alcança e quais vértices fazem parte da fila...

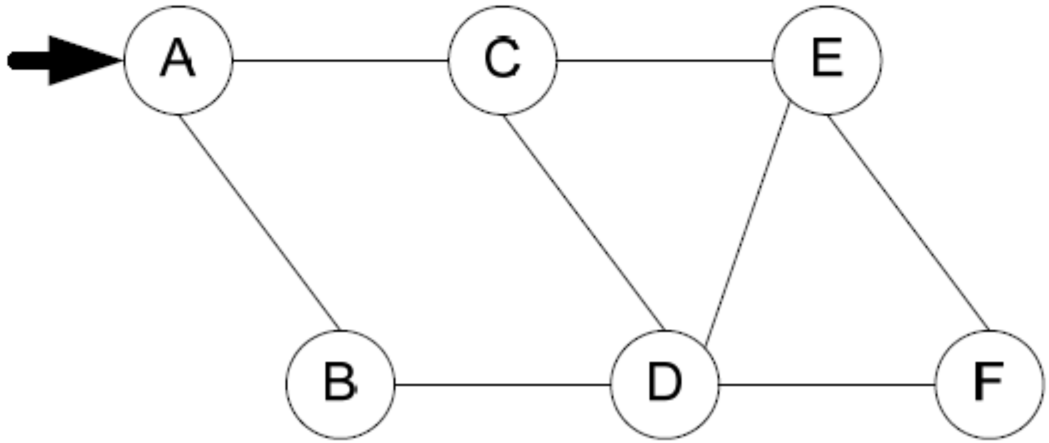
- Partindo de A até G

- Partindo de A até I



Exercício [poscomp 2009]

Considere o algoritmo de busca em largura em grafos. Dado o grafo a seguir e o vértice A como ponto de partida, a ordem em que os vértices são descobertos é dada por:



A) A B C D E F
B) A B D C E F
C) A C D B F E
D) A B C E D F
E) A B D F E C