

Grafos: Busca

Algoritmos e Estruturas de Dados 2

Graça Nunes

Percorrendo um grafo

Percorrendo um Grafo

- ❑ Percorrer um grafo é uma **tarefa fundamental**
- ❑ Pense no caso de se procurar uma certa informação associada a um vértice/aresta num grafo
- ❑ Deve-se ter uma forma **sistemática** de visitar as arestas e os vértices
- ❑ O algoritmo deve ser suficientemente **flexível** para se adequar à diversidade de grafos

Eficiência

Percorrendo um Grafo

- Eficiência

- Não deve haver repetições (desnecessárias) de visitas a um vértice e/ou aresta

Correção

Percorrendo um Grafo / Busca em Grafos

- Correção
 - Todos os vértices e/ou arestas devem ser visitados, se o objetivo for passar por todos

Algoritmo Básico de Busca em Grafo

- Utiliza o conceito de marcar os vértices, de modo a registrar que ele já foi visitado.
- Seja G um grafo conexo em que todos vértices não estão marcados (não foram ainda visitados)
- Passo Inicial:
 - escolher e marcar um vértice arbitrário v ;
- Passo Geral:
 - selecionar (explorar) uma aresta (v,w) incidente a um vértice marcado v e que não tenha sido selecionada anteriormente
 - Se w é não marcado, marca-se w
- O processo termina quando todas as arestas de G tiverem sido selecionadas

Algoritmo Geral de Busca num Grafo Conexo

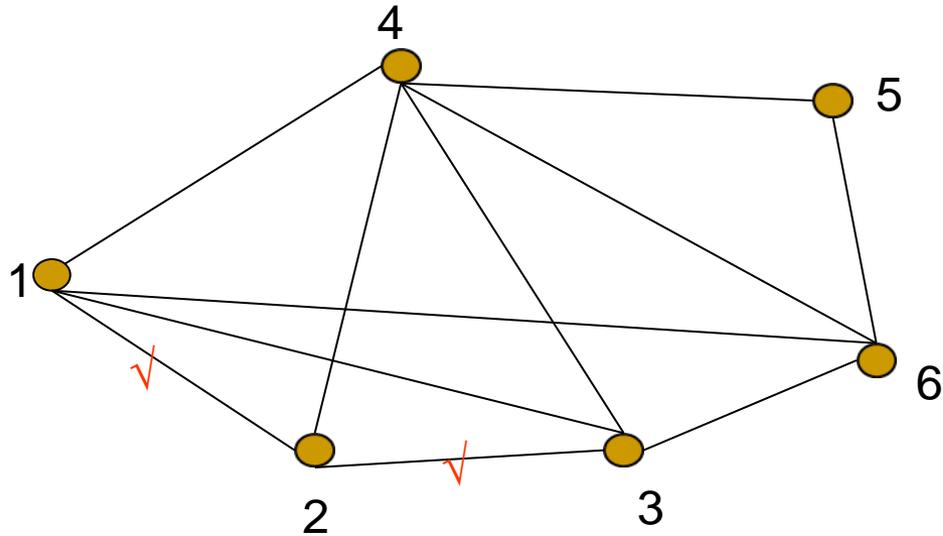
- Dado um Grafo (V,A) conexo:

início

escolher e marcar um vértice inicial;
enquanto existir algum vértice v marcado e
incidente a uma aresta (v,w) não explorada,
faça escolher o vértice v e explorar
(marcar) a aresta (v,w)
se w é não marcado
então marcar w

fim

Exemplo



Vértice inicial: 1 (raiz da busca)

Marca (v) Explora (aresta)

1 (1,2)

2 (2,3)

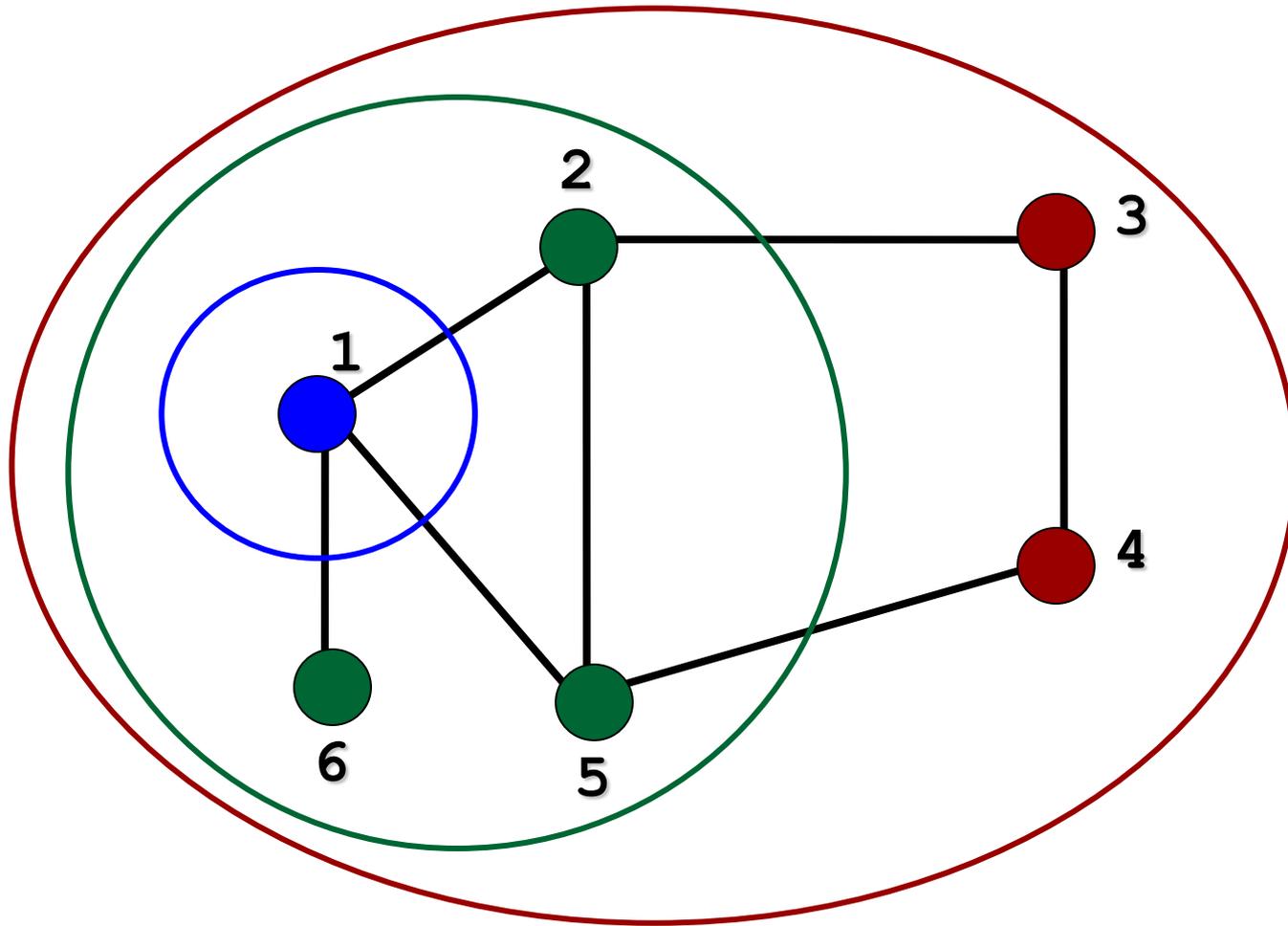
.....

Variação do Algoritmo Geral

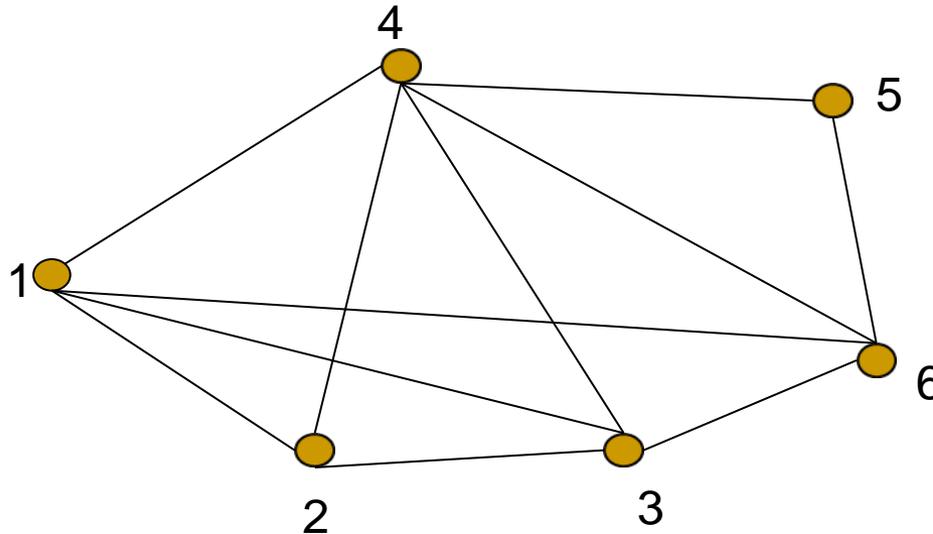
- ❑ BFS – *Breadth-First Search – Busca em Largura*
- ❑ A busca em largura é obtida do método básico, onde a seleção do próximo vértice marcado obedece a:
 - *Dentre todos os vértices marcados e incidentes a alguma aresta ainda não explorada, escolher aquele menos recentemente alcançado na busca*
- ❑ Dessa forma, os vértices são armazenados numa fila de modo a serem processados “*first in first out*”

BFS – exemplo

Percorre-se o grafo como se houvesse uma onda na água!



Aplique o algoritmo ao grafo abaixo



Listas de Adjacências:

1: (4,2,3,6)

2: (1,4,3)

3: (2,1,4,6)

4: (1,2,3,6,5)

5: (4,6)

6: (3,1,4,5)

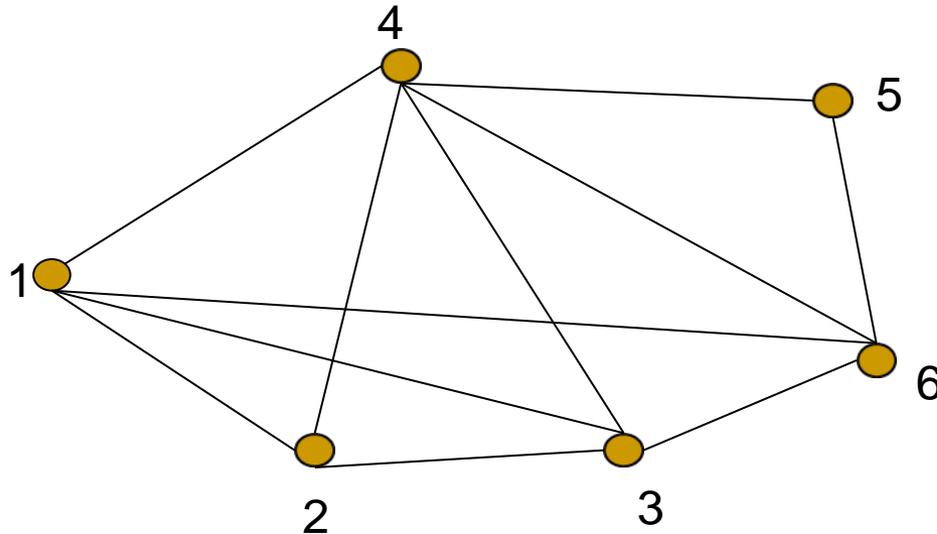
Vértice inicial: 1 (raiz da busca)

Q:

Vértices Marcados:

Arestas Visitadas:

Aplique o algoritmo ao grafo abaixo



Listas de Adjacências:

1: (4,2,3,6)

2: (1,4,3)

3: (2,1,4,6)

4: (1,2,3,6,5)

5: (4,6)

6: (3,1,4,5)

Vértice inicial: 1 (raiz da busca)

Q: (1,4,2,3,6,5)

Vértices Marcados: 1,2,3,6,5

Arestas Visitadas: (1,4) (1,2) (1,3) (1,6) (4,2) (4,3) (4,6) (4,5) (2,3) (3,6) (6,5)

Algoritmo Busca em Largura

Dado $G(V,A)$, conexo:

escolher uma raiz s de V

definir uma fila Q , vazia

marcar s

inserir s em Q

enquanto Q não vazia faça

seja v o 1o. vértice de Q

para cada $w \in \text{ListaAdjacencia}(v)$ faça

se w é não marcado então

(I) visitar (v,w)

marcar w

inserir w em Q

(II) senão se $w \in Q$ então visitar (v,w) /* w alcançado por outro caminho*/

/*senão já processou w e portanto (w,v) */

/*fim_para*/

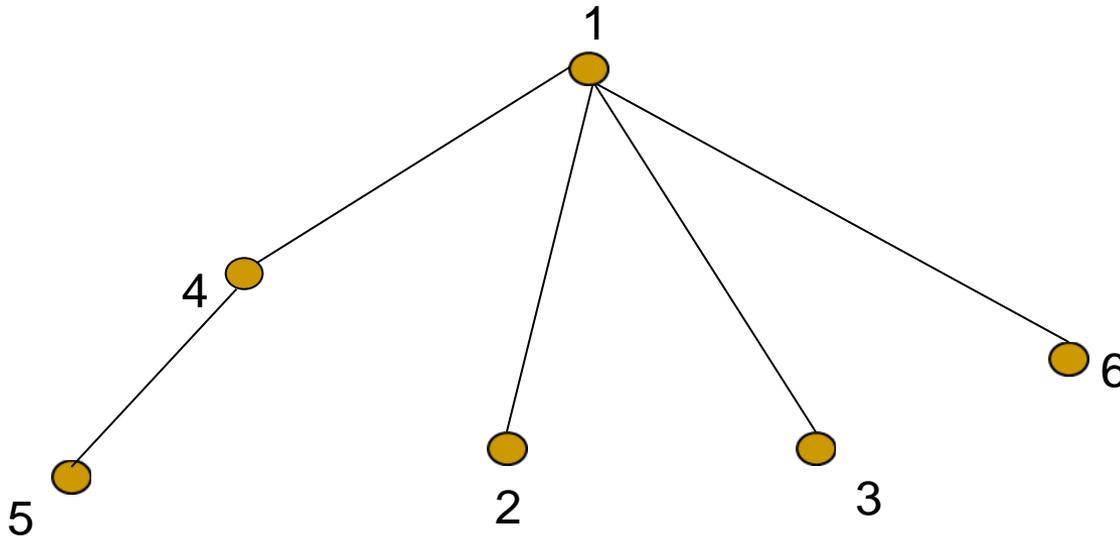
retirar v de Q

/*fim_enquanto*/



Árvore Geradora do Grafo

Seja E_T o conjunto das arestas visitadas em (I). O Grafo $T(V, E_T)$ é uma árvore geradora de G (também chamada de árvore de largura de G).



BFS (Busca em Largura)

- BFS – *Breadth-First Search*
 - Todos os nós com distância k a um nó v são visitados antes dos nós com distância $k+1$ (garantido pelo uso da fila)
 - Descobre todos os vértices alcançáveis a partir de v (portanto, pode ser usada para achar caminhos)
 - A busca em largura resulta no **caminho mais curto** entre o vértice inicial e um vértice qualquer x

BFS – outra forma de visualizar

- ❑ É comum a utilização de esquemas de cores para identificar os nós ainda não visitados (**branco**), visitados (**cinza**) e já completamente processados (**preto**)
- ❑ Entre todos os visitados, o próximo a ser processado é o primeiro de uma Fila (Fila_Visitado)

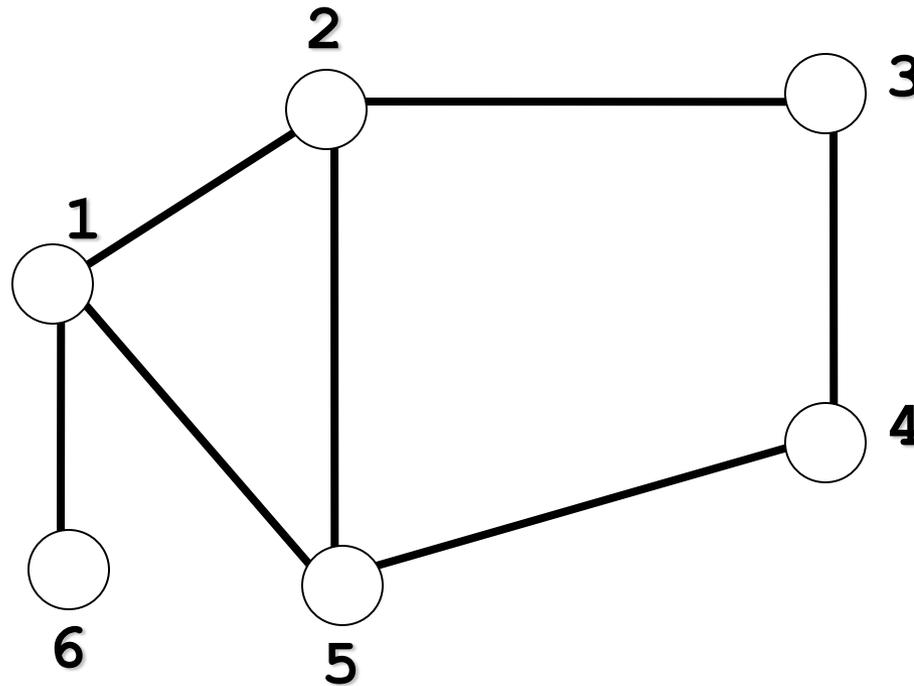
BFS (Busca em Largura)

Percorrendo um Grafo

- BFS – *Breadth-First Search*
 - Todos os vértices são inicializados **brancos**
 - Quando um vértice v é descoberto pela primeira vez, ele se torna **cinza**
 - Quando todos os vértices adjacentes a v são descobertos, v se torna **preto**

BFS – exemplo

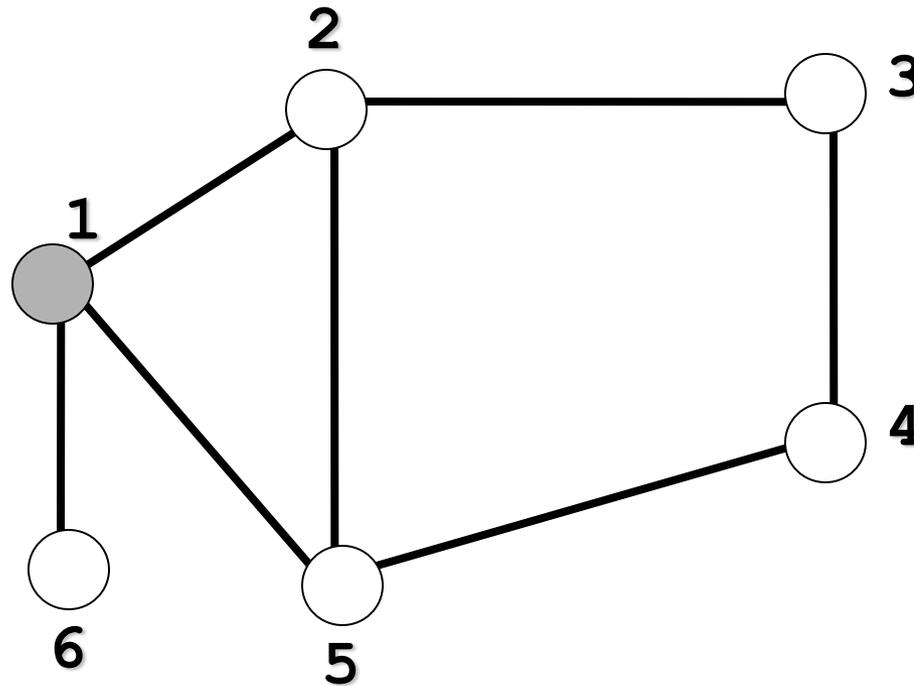
Percorrendo um Grafo: BFS



NãoVisitado:[1,2,3,4,5,6]; Processado:[]; Fila_Visitado:[];

BFS – exemplo

Percorrendo um Grafo: BFS

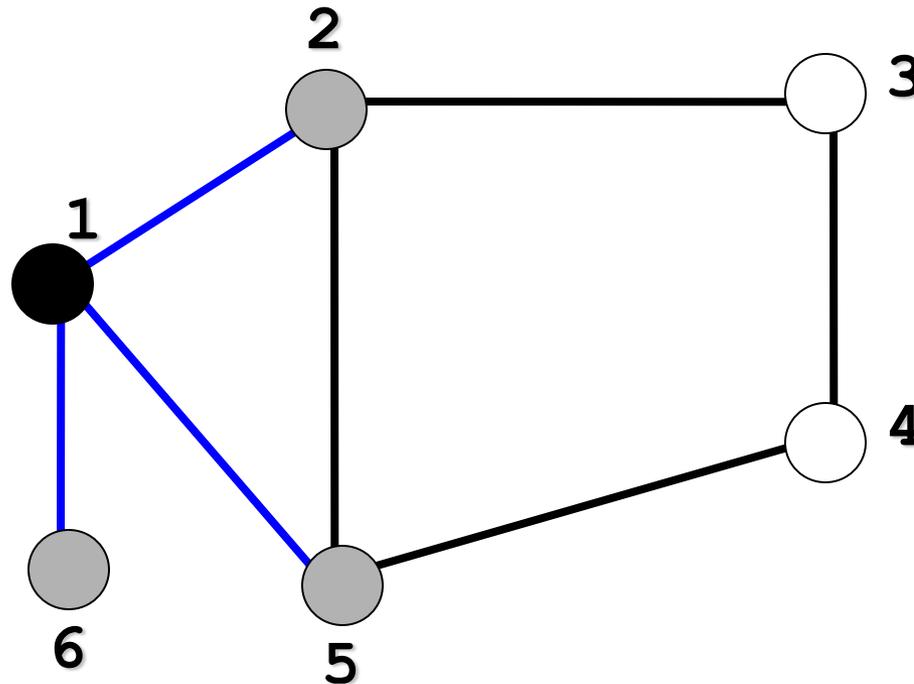


inicial: 1; Fila_Visitado:[1]; NãoVisitado:[2,3,4,5,6]; Processado:[];

BFS – exemplo

Percorrendo um Grafo: BFS

K=1

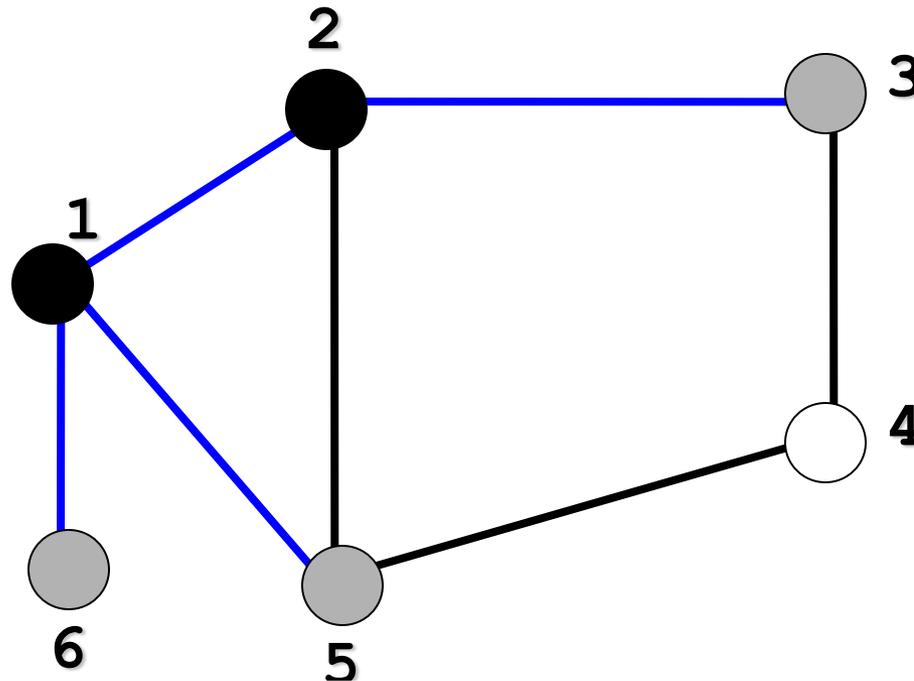


Visitam-se todos os nós não visitados adjacentes a 1: 2, 5 e 6;
NãoVisitado:[3,4]; Processado:[1]; Fila_Visitado:[2,5,6]

BFS – exemplo

■ Percorrendo um Grafo: BFS

K=2

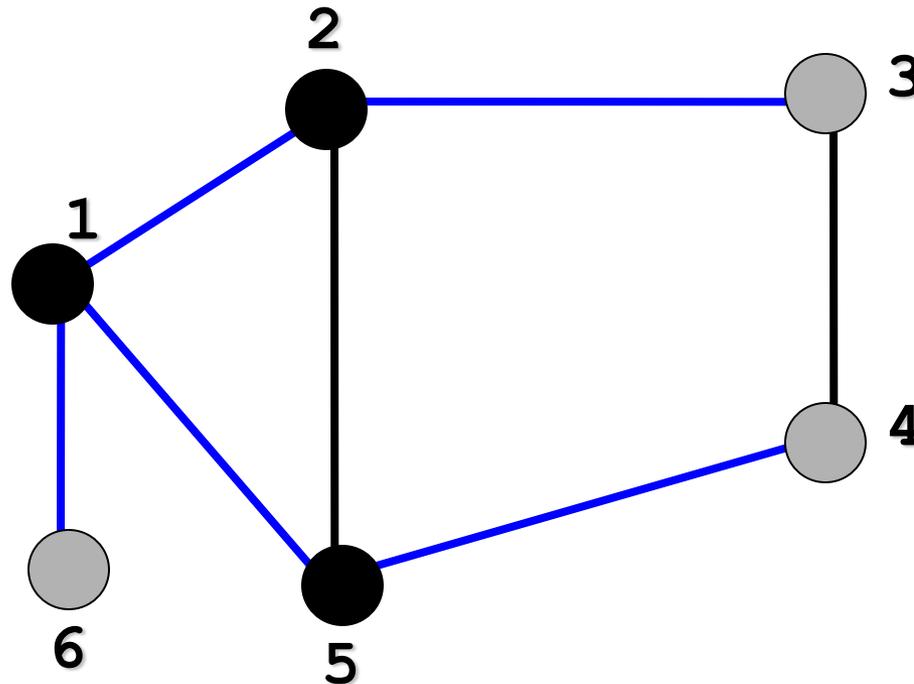


Visitam-se todos os nós não visitados adjacentes a 2: 3;
NãoVisitado:[4]; Processado:[1,2]; Fila_Visitado:[5,6,3]

BFS – exemplo

Percorrendo um Grafo: BFS

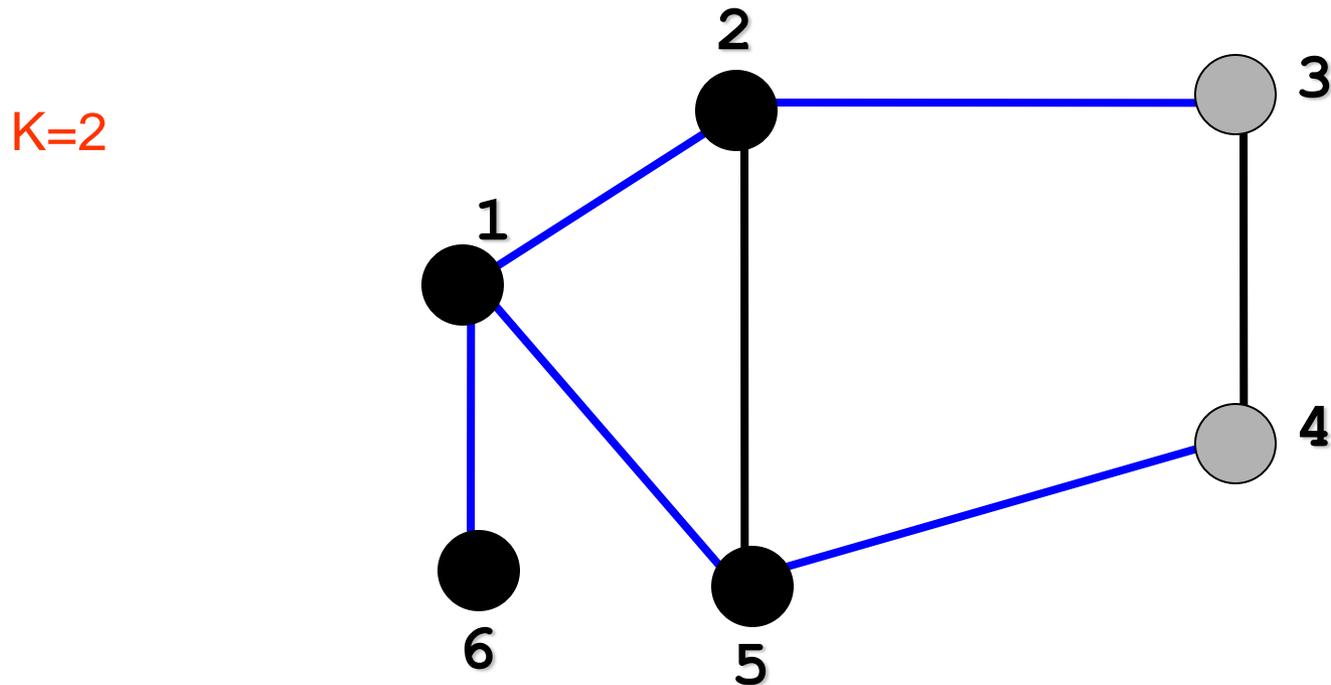
K=2



- Visitam-se todos os nós não visitados adjacentes a 5: 4
NãoVisitado:[]; Processado:[1,2,5]; Fila_Visitado:[6,3,4]

BFS – exemplo

Percorrendo um Grafo: BFS

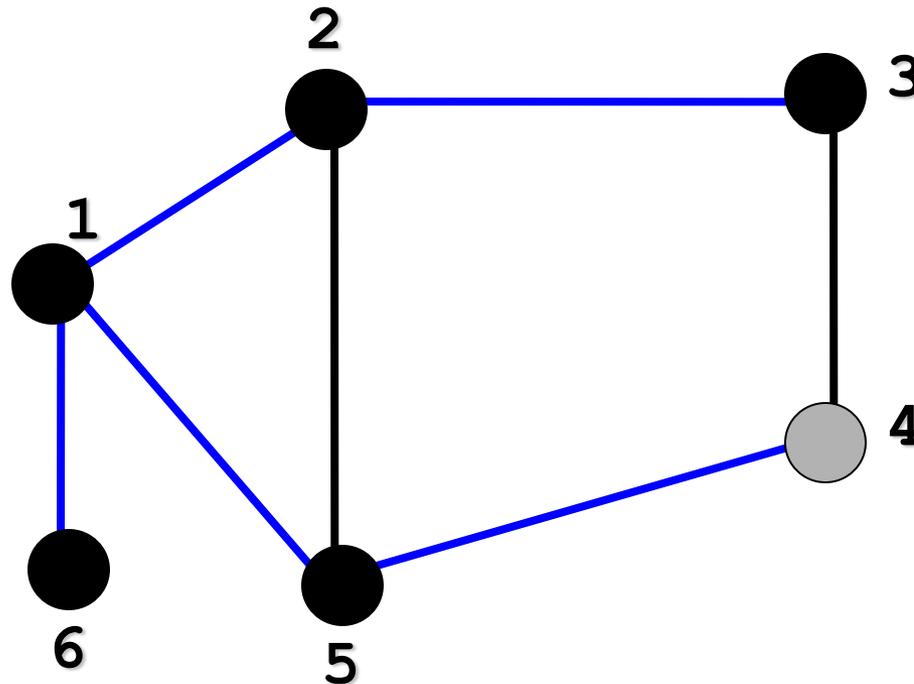


Visitam-se todos os nós não visitados adjacentes a 6: nenhum;
NãoVisitado:[]; Processado:[1,2,5,6]; Fila_Visitado:[3,4]

BFS – exemplo

Percorrendo um Grafo: BFS

K=3

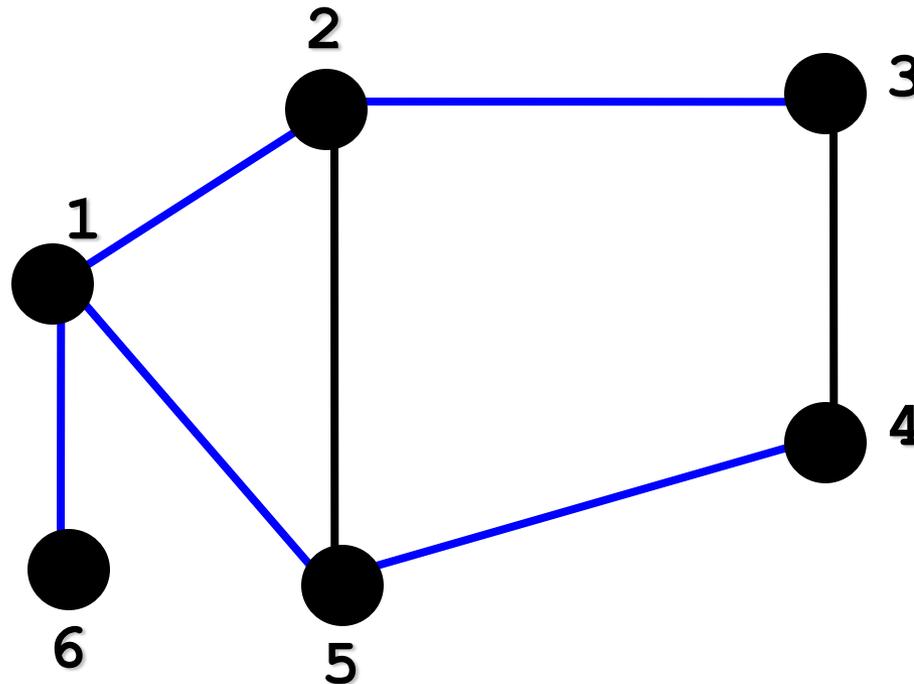


Visitam-se todos os nós não visitados adjacentes a 3: nenhum;
NãoVisitado:[]; Processado:[1,2,5,6,3]; Fila_Visitado:[4]

BFS – exemplo

■ Percorrendo um Grafo: BFS

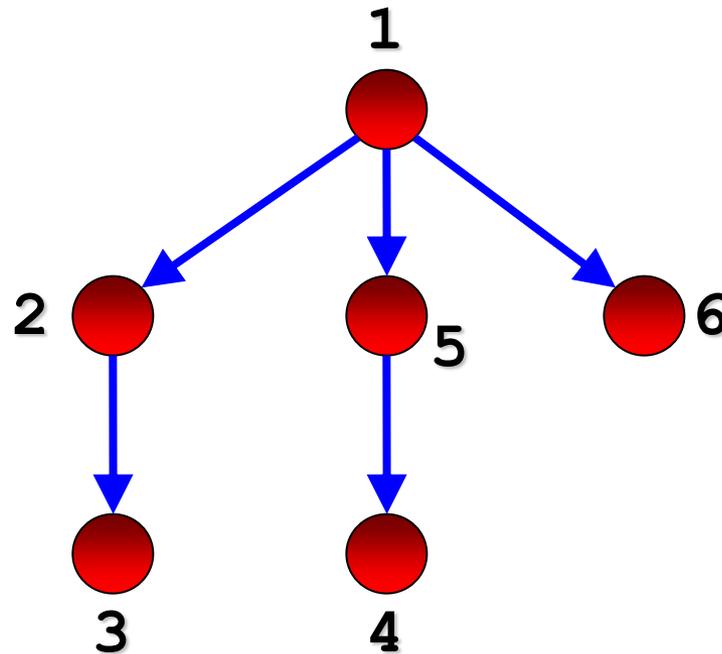
K=3



Visitam-se todos os nós não visitados adjacentes a 4: nenhum;
NãoVisitado:[]; Processado:[1,2,5,6,3,4]; Fila_Visitado:[]

BFS – exemplo

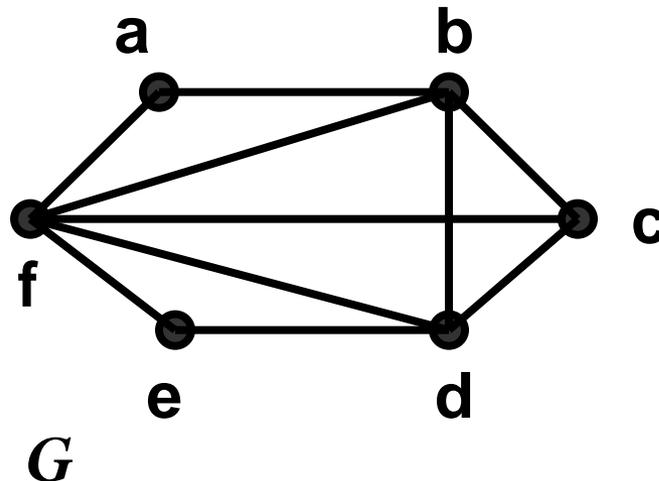
Percorrendo um Grafo: **árvore de busca em largura**



Atenção: arestas (2,5) e (3,4) não percorridas!

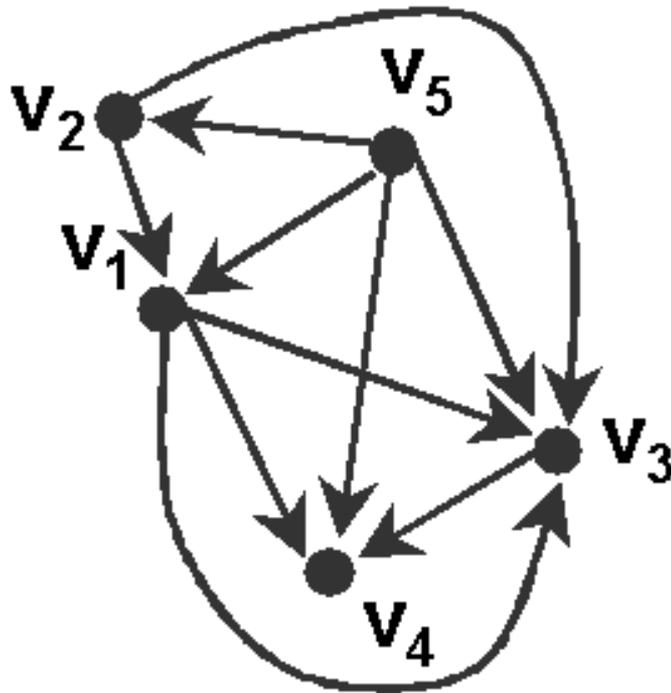
BFS (Busca em Largura)

- **Exercício:** faça a busca em largura no grafo abaixo, mostrando a ordem de visita aos vértices



BFS (Busca em Largura)

- **Exercício:** faça a busca em largura no dígrafo abaixo, mostrando a ordem de visita aos vértices



BFS (Busca em Largura)

- Implementação da busca em largura com Listas de Adjacência

Complexidade do BFS

$O(|V| + |E|)$, ou seja, linear em relação ao tamanho da representação do grafo por listas de adjacências

- Todos os vértices são enfileirados/desenfileirados no máximo uma vez; o custo de cada uma dessas operações é $O(1)$, e elas são executadas $O(|V|)$ vezes
- A lista de adjacências de cada vértice é percorrida no máximo uma vez (quando o vértice é desenfileirado); o tempo total é $O(|E|)$ (soma dos comprimentos de todas as listas, igual ao número de arestas)