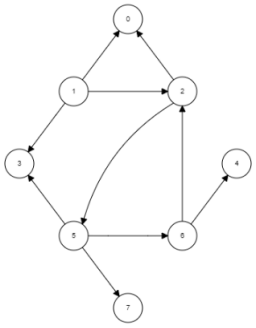
# Lista de Exercícios 6

Universidade Federal do Ceará - Campus Quixadá
Projeto e Análise de Algoritmo — QXD0041 – 2023.2
Prof. Fabio Dias

## Busca em Grafos

1. Aplique a Busca em Largura no grafo abaixo, mostrado a cada passo, o crescimento da árvore de busca em largura.




2. Modifique a Busca em Largura para que, além de calcular o tamanho do menor caminho da origem a todo vértice, ela também calcule o número de caminhos distintos de menor tamanho.

3. Em muitas aplicações de redes de computadores, são necessário definir redundâncias na rede, para evitar que um certo servidor fique fora do ar devido a uma falha em um link. Dado um grafo $G = (V, E)$ conexo e não direcionado, precisamos calcular a quantidade de caminhos distintos de um vértice origem a todos os demais vértices do grafo. Modifique a Busca em Largura para resolver esse problema.

4. Modifique a Busca em Largura para funcionar em um grafo representado por uma matriz de adjacências. Qual é o tempo de execução dessa versão?

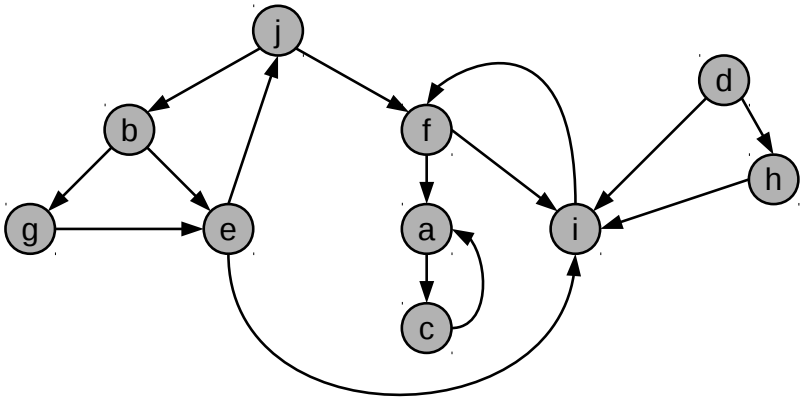
5. Dado um caminho entre dois vértices de um grafo, o tamanho do caminho em um grafo não ponderado é a quantidade de arestas do caminho. A Busca em Largura encontra o caminho mínimo do vértice origem $s$ até cada vértice que pode ser alcançado por ele em um grafo. Defina a **distância do caminho mínimo** $\delta(s, v)$ de $s$ até $v$ como o número mínimo de arestas em qualquer caminho do vértice $s$ ao vértice $v$. Se não há nenhum caminho entre os dois vértices, então $\delta(s, v) = \infty$.

   Dito isso, seja $T = (V, E)$ uma árvore. O **diâmetro** de uma árvore $T$ é definido por $\max_{\forall u,v \in V} \delta(u, v)$ isto é, a maior de todas distâncias de caminhos mínimos na árvore. Dê um algoritmo eficiente para calcular o diâmetro de uma árvore e analise o tempo de execução de seu algoritmo.

6. Na busca em Largura, cada vértice é pintado de cinza imediatamente antes de ser adicionado na Fila $Q$, indicando que esse vértice foi visitado. O que acontece com a busca em Largura se pintamos o vértice de cinza apenas quando o vértice é removido da Fila $Q$? O algoritmo ainda funciona? Sua complexidade é alterada?

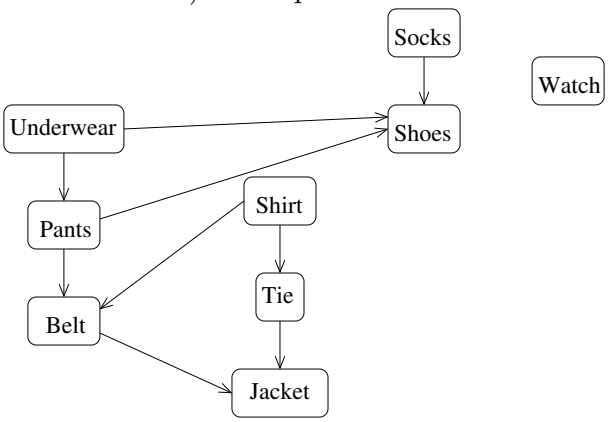
7. Dado um grafo não direcionado, como podemos determinar se esse grafo possui ou não um ciclo no tempo $O(V + E)$?

8. Mostre como a busca em profundidade funciona no grafo da figura abaixo. Presuma que o laço da rotina principal DFS considera os vértices em ordem alfabética e presuma que cada lista de adjacências está ordenada alfabeticamente. Mostre o tempo de descobrimento (**d**) e o tempo de término (**f**) de cada vértice na caixa abaixo.

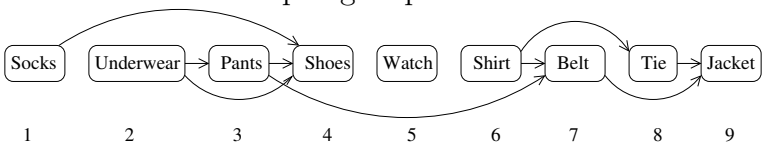



| | a | b | c | d | e | f | g | h | i | j |
|---|---|---|---|---|---|---|---|---|---|---|
| **d** | | | | | | | | | | |
| **f** | | | | | | | | | | |

9. Uma ordenação topológica de um grafo direcionado acíclico é uma ordem linear dos vértices tal que $\forall(u, v) \in E \Rightarrow u$ aparece antes $v$ na ordem linear. Normalmente é usada em problema de agendamento e similares. Por exemplo, ordem de Roupas (a seta implica "deve vestir antes"). Nós queremos obter a ordem que de vestir.




A ordem abaixo é uma ordem topológica possível:




Implemente um algoritmo de complexidade $O(V+E)$ que dado um grafo direcionado

acíclico, devolve a ordenação topológica do grafo.

10. Implemente uma busca em profundidade (DFS) usando uma pilha (de forma a eliminar a recursão). O seu algoritmo deverá *devolver uma floresta de busca em profundidade* representada por um vetor $\pi$ e deve executar em tempo $O(V + E)$. Você pode utilizar as sub-rotinas de uma pilha como caixas-pretas: CriarPilha($Q$), Topo($Q$), Desempilhar($Q$), Empilhar($Q$, $v$).