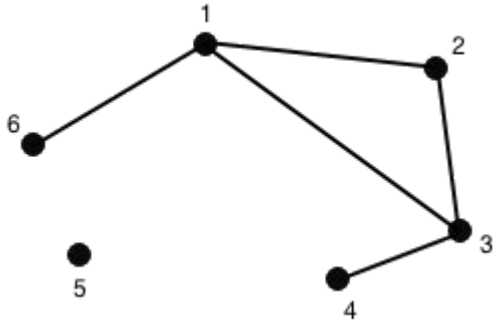
Grafos

O que são grafos

De forma resumida um grafo é um conjunto de vértices V e um conjunto de arestas E que liga um par de vértices (De forma mais informal, um conjunto de pontos (vértices) e traços (arestas)).

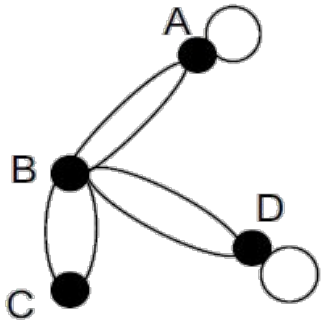


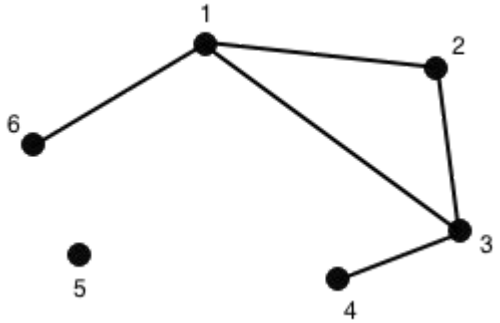
Para que serve grafos

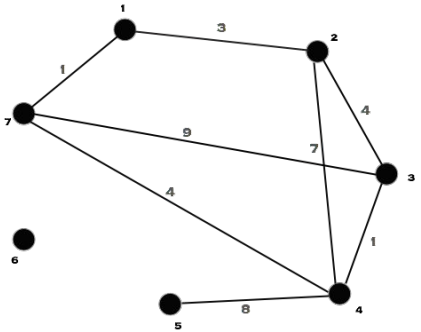
- Problema 1: Verificar se existe um caminho entre duas cidades. (os vértices são as cidades e as arestas representam as estradas/caminhos).
- Problema 2: Em um tabuleiro de xadrez você quer verificar se é possível mover uma peça até uma casa, ex: um cavalo até outra casa (os vértices são as casas do tabuleiro, e existe uma aresta entre duas casas se é possível ir de uma até outra usando o movimento de uma determinada peça).
- Problema 3: Descobrir o menor caminho entre dois pontos. (Os pontos são os vértices e o caminho as arestas).

Definições

- O grau de um vértice é o número de arestas que são incidentes nele.
- Dizemos que dois vértices são vizinhos se existe uma aresta que conecta os dois.
- É dito que o grafo tem peso se suas arestas possuem pesos.
- Uma aresta é chamada de self-loop se é da forma (u, u) — ou seja, liga um vértice a si mesmo.
- Um grafo é chamado de multi-grafo se possui duas ou mais arestas ligando um mesmo par de vértices.
- Um grafo é dito simples se não é um multi-grafo e não possui self-loops.
- Em grafos não direcionados a aresta (u, v) equivale a aresta (v, u).







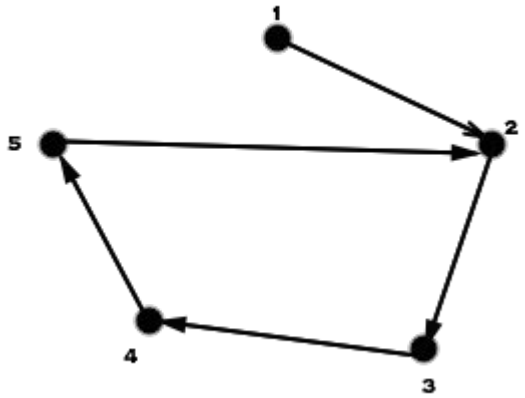
Tipos de grafos

Tipos mais comuns de grafos:

- Grafo direcionado;
- Grafo completo;
- Grafo Conexo;
- Grafo Ciclo;
- Grafo Árvore;

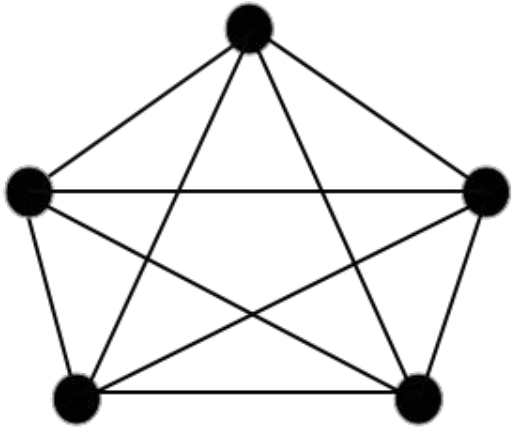
Grafo Direcionado

Um grafo direcionado é quando suas arestas só podem ser percorridas em um sentido.



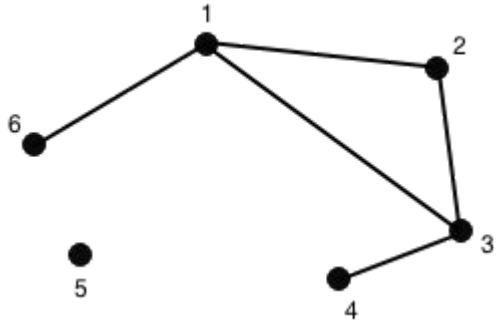
Grafo Completo

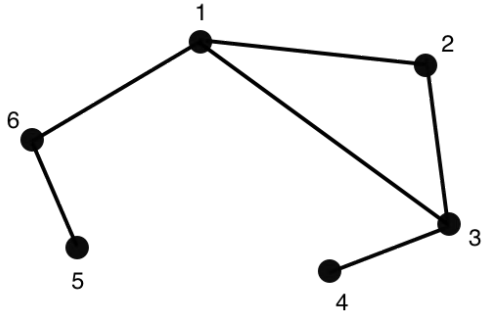
Um grafo completo é um grafo onde todos os vértices estão conectados por arestas entre si. (Cada vértice está ligado com todos os outros).



Grafo Conexo

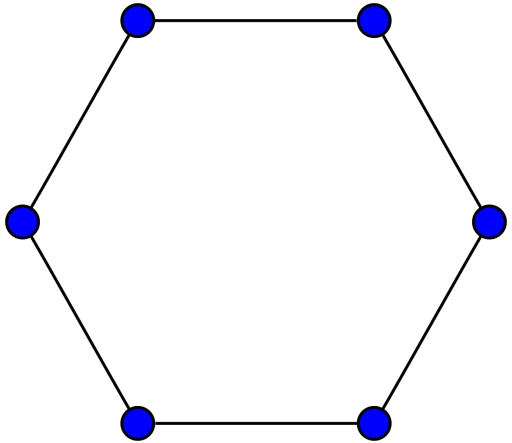
Um grafo conexo é um grafo onde cada um dos seus vértices tem pelo menos uma aresta ligando ele a outro. Grafos conexos são bidirecionais (é possível passar em ambos os sentidos em uma aresta).





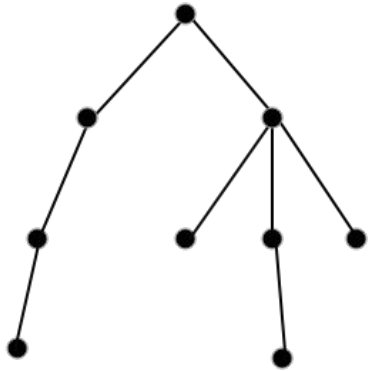
Ciclo

Um Ciclo em um grafo ocorre quando forma-se uma espécie de círculo no grafo de forma que se eu sair de um vértice eu consigo voltar para ele sem passar duas vezes em um vértice.



Grafo Árvore

Um grafo bidirecional é chamado árvore quando não ocorre nenhum ciclo dentro do mesmo.



Formas de representar um grafo.

Para utilizar um grafo em nosso programa, precisamos representar ele de alguma forma. Existem várias formas de representar um grafo, vamos ver as 3 mais comuns.

- Matriz de Adjacência
- Lista de Adjacência
- Representação Implícita

Matriz de Adjacência

A matriz de adjacência consiste, para cada par de vértices, saber se existe ou não uma aresta entre eles. guardamos na posição (i,j) da matriz se existe uma aresta entre i e j, colocamos 1 se existir, e 0 se não, se a aresta for bidirecional coloco que ela existe na posição (i, j) e (j, i);

	V_1	V_2	V_3	V_4	V_5	V_6
V_1	0	1	1	0	0	1
V_2	1	0	1	0	0	0
V_3	1	1	0	1	0	0
V_4	0	0	0	0	0	0
V_5	0	0	0	0	0	0
V_6	1	0	0	0	0	0

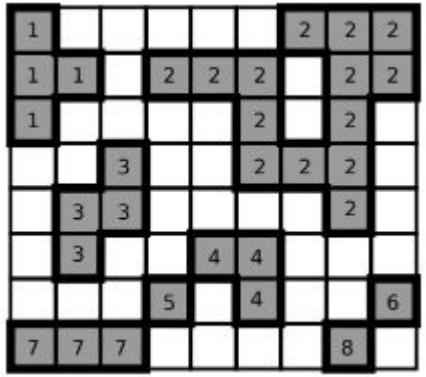
Lista de Adjacência

A lista de adjacência baseia - se em guardar para cada vértice a lista de seus vizinhos, para falar que existe uma aresta entre A e B basta adicionar B na lista de vizinhos de A, e se o grafo for bidirecional também adiciono A na lista de B.

Vértice	Vizinhos
1	{2, 3, 6}
2	{1, 3}
3	{1, 2}
4	{3}
5	{}
6	{1}

Representação Implícita

Em alguns casos não é preciso salvar nenhuma aresta para poder se trabalhar com um grafo. Por exemplo, imagine que você está trabalhando com um cavalo em um tabuleiro de xadrez. Dado um determinado vértice (a posição no tabuleiro), já se pode saber quais são os vizinhos dele, sendo esses os vértices que representam as 8 possíveis casas para as quais o cavalo pode se mover (portanto que não saia do tabuleiro, claro).



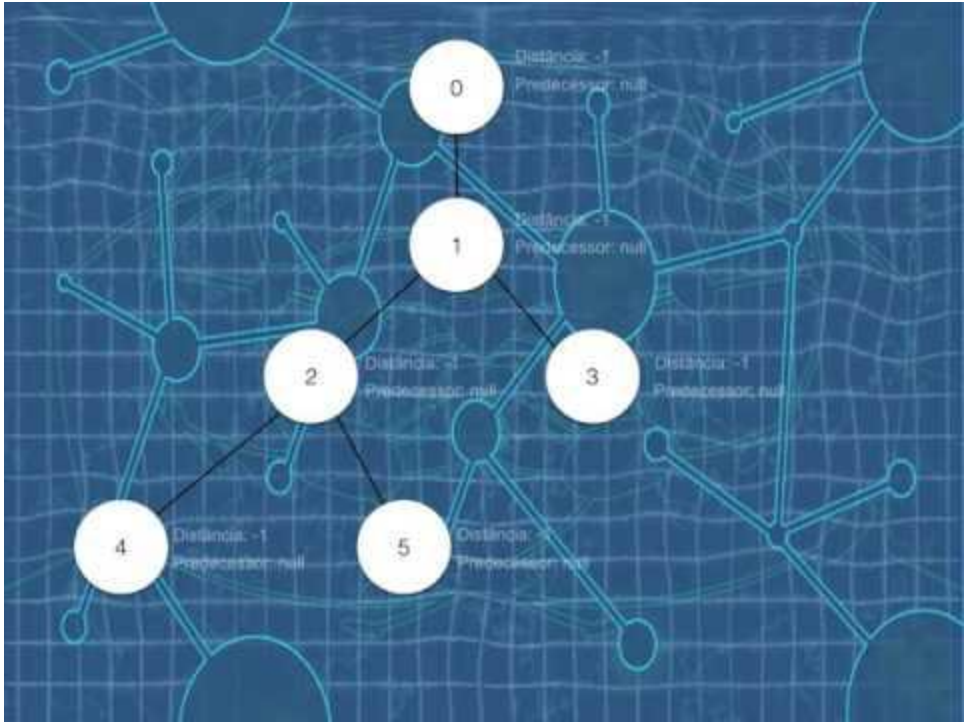
1 0 0 0 0 0 1 1 1
1 1 0 1 1 1 0 1 1
1 0 0 0 0 1 0 1 0
0 0 1 0 0 1 1 1 0
0 1 1 0 0 0 0 1 0
0 1 0 0 1 1 0 0 0
0 0 0 1 0 1 0 0 1
1 1 1 0 0 0 0 1 0

Busca em grafos

Em alguns casos teremos que realizar buscas em grafos, seja para saber se é possível ir de A até B ou marcar os vizinhos de um vértice. Existem 2 algoritmos para esse problema: Busca em profundidade (DFS) e busca em largura (BFS).

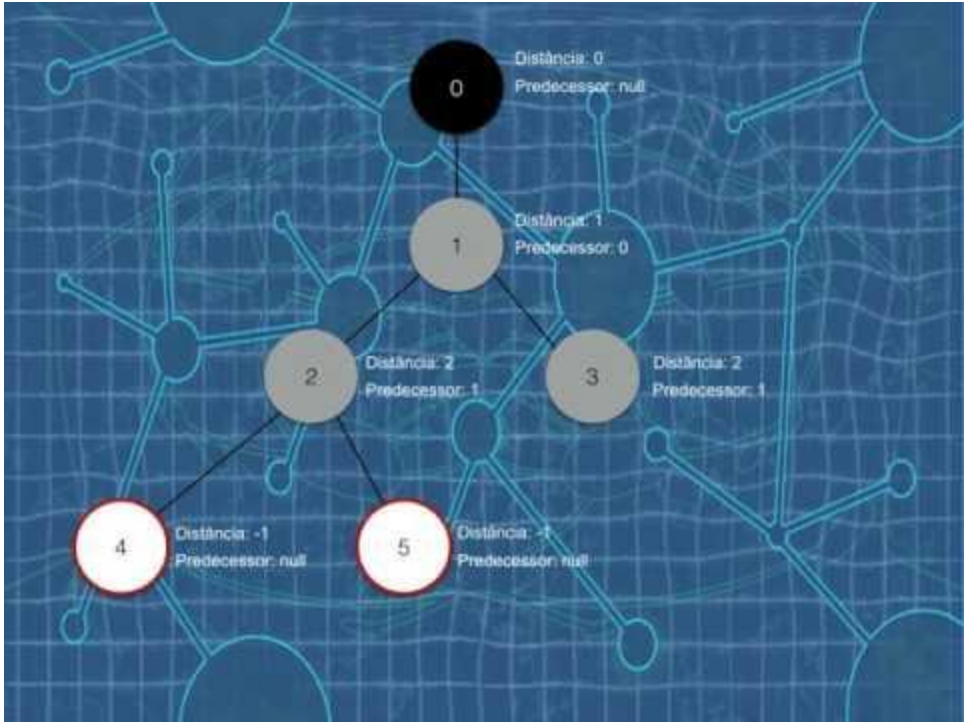
Busca em profundidade (DFS)

A DFS consiste em cada passo olhar os vizinhos do vértice V que está sendo avaliado, e depois os vizinhos do vizinho, sucessivamente.


0
Distância: -1
Predecessor: null
1
2
Distância: -1
Predecessor: null
3
Distância: -1
Predecessor: null
4
Distância: -1
Predecessor: null
5

Busca em largura (BFS)

Se trata de fazer o mesmo processo da DFS, porém em vez de chamar uma função recursivamente em um vizinho, este é colocado em uma fila e processado depois.


0
Distância: 0
Predecessor: null
1
Distância: 1
Predecessor: 0
2
Distância: 2
Predecessor: 1
3
Distância: 2
Predecessor: 1
4
Distância: -1
Predecessor: null
5
Distância: -1
Predecessor: null

Exercícios:

- DFS
  - Toca do Saci
  - Manchas de pele
  - Fissura perigosa
  - Câmeras
  - Sr. Sapo
  - Dona formiga (Maior número de salões)
  - Pandemia
  - Chuva
- BFS
  - Coloração de Cenários de Jogos
  - Pulo do Gato (P1)
  - Gincana
  - Mapa
  - Lá vai tinta!

Links Úteis

- MaratonUSP: https://www.youtube.com/c/MaratonUSP
- Neps Academy: https://neps.academy/br/course/algoritmos-em-grafos-(codcad)/
- NOIC: https://noic.com.br/materiais-informatica/curso/graphs-01/
- Maratona UFMG: https://www.youtube.com/channel/UCWNq1dCrk4eJFZEDQhrvTbg
- Graph Editor: https://csacademy.com/app/graph_editor/
- VisualGO - Grafos: https://visualgo.net/en/graphds
- VisualGO - Algoritmos: https://visualgo.net/en/dfsbfs